Hochschule Mittweida (FH)
University of Applied Sciences

MITTWEIDA
ROSSWEIN

Fachbereich Medien

Jens Kabisch

Evaluierung von Adobe Flex 3 am Beispiel einer Anwendung zur statistischen Auswertung von Werbespielen

- Diplomarbeit -

Hochschule Mittweida - University of Applied Science (FH)

Mittweida - 2009

Hochschule Mittweida (FH)
University of Applied Sciences

MITTWEIDA
ROSSWEIN

Fachbereich Medien

Jens Kabisch

Evaluierung von Adobe Flex 3 am Beispiel einer Anwendung zur statistischen Auswertung von Werbespielen

- eingereicht als Diplomarbeit -

Hochschule Mittweida - University of Applied Science (FH)

vorgelegte Arbeit wurde verteidigt am:

Erstprüfer	Zweitprüfer
Prof. Dr. phil. Ludwig Hilmer	Dipl.-Ing. Sieglinde Klimant

Mittweida - 2009

Bibliographische Bezeichnung:

Jens Kabisch: Evaluierung von Adobe Flex 3 am Beispiel einer Anwendung zur statistischen Auswertung von Werbespielen“, Diplomarbeit, 101 S.; Hochschule Mittweida, Fachbereich Medien (2009)

Referat:

In dieser Diplomarbeit wird Adobe Flex 3 hinsichtlich seiner Bedeutung als Entwicklungsframework für Rich-Internet-Applications untersucht. Um eine aussagekräftige Analyse der Adobe Flex 3 Technologie durchführen zu können, werden zunächst bereits etablierte Rich-Internet-Application-Frameworks und deren Besonderheiten vorgestellt.

Im Hauptteil der Arbeit wird das Adobe Flex 3 Framework genauer betrachtet und analysiert. Dabei wird zum einen auf die verschiedenen Software-Komponenten eingegangen, aus denen das Framework besteht. Zum anderen werden die bedeutendsten MXML- und ActionScript 3 Bibliotheken betrachtet, die im Flex Software-Developer-Kit mitgeführt werden. Um eine kleinen Einblick in die Flex-Programmierung zu erhalten, werden zusätzlich noch Quellcode-Beispiele für die Verwendung der Klassen gezeigt.

Außerdem wurde im Rahmen dieser Arbeit eine Flex-Anwendung erstellt, mit der es möglich ist, statistische Daten von Werbespielen zu betrachten bzw. auszuwerten. Diese Applikation wird im letzten Teil der Arbeit vorgestellt, die Entwicklung der Anwendung erläutert und letztendlich die Eignung von Adobe Flex 3 für die Lösung der gegebenen Softwareanforderungen eingeschätzt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Einleitung

1.1 Motivation

Das World Wide Web (kurz WWW) wurde im Jahre 1989 am CERN (Europäische Organisation für Kernforschung) von Tim Berners-Lee entwickelt[1]. Es sollte ursprünglich dazu dienen, Forschungsergebnisse auf einfache Art und Weise auszutauschen. Die wesentliche Grundlage des WWW stellt dabei die Hypertext Markup Language (kurz HTML) dar. HTML ist eine textbasierte Auszeichnungssprache und ermöglicht es Inhalte wie Texte, Bilder und Hyperlinks zu anderen Seiten innerhalb eines Dokumentes zu strukturieren. Interpretiert und dargestellt werden HTML Dokumente mit Hilfe eines Webbrowsers. Mit der Entwicklung und Veröffentlichung des ersten grafikfähigen Browsers Mosaic im Jahre 1993 durch das National Center for Supercomputing Applications bekam das WWW einen enormen Schub. Bereits am Ende des gleichen Jahres waren etwa 2 Millionen Kopien des Browsers im Umlauf[2].

Das World Wide Web wuchs in den folgenden Jahren, gemessen an der Anzahl der zur Verfügung stehenden Hosts, exponentiell an. Immer größere Teile der nicht-akademischen Bevölkerung nutzten inzwischen das Internet und trieben den Ausbau der Internet-Infrastruktur voran. Mit wachsender Popularität des Internets wurden auch kommerzielle Unternehmen auf die neue Technologie aufmerksam und begannen eigene Websites aufzubauen, um ihre Produkte zu präsentieren. Mit der Zeit wuchsen die Möglichkeiten und zugleich auch die Ansprüche an die Internet-Technologie. So wurde beispielsweise das HTML-Dokumentenformat ständig weiterentwickelt. Konnte man in den ersten HTML-Versionen lediglich Bilder in ein Dokument integrieren, kann man mittlerweile nahezu alle Dateitypen in eine Website einbetten oder verlinken. Durch sogenannte Plugins oder Ergänzungsmodule ist es mit modernen Browsern auch möglich, verschiedenste Multimediainhalte, angefangen von einfachen Animationen bis hin zu Musik und Videos darzustellen.

Seit der Entwicklung des World Wide Web und vor allem seit dem ersten grafikfähigen Webbrowser Mosaic hat das Internet einen rasanten Aufstieg erlebt. Dank dieser Technologien

1 vgl. The original proposal of the WWW, http://www.w3.org/History/1989/proposal.html, aufgerufen am 8. März 2008

2 vgl. Geschichte des Internets, http://www.netplanet.org/geschichte/neunziger.shtml, aufgerufen am 8. März 2008

konnten nun auch Laien auf das Netz zugreifen. Eine wachsende Zahl von Nutzern führte zu vielen kommerziellen Angeboten im WWW. Doch sowohl die Ansprüche der Nutzer als auch die der Website-Entwickler sind, im Hinblick auf die Einbettung multimedialer Inhalte, ebenfalls gestiegen.

Im Mai 1996 brachte die kleine amerikanische Firma FutureWave Software das vektorbasierte Animationswerkzeug FutureSplash Animator auf den Markt[3]. Mit Hilfe eines kostenlosen Browser-Plugins konnten FutureSplash-Animationen auch in Browsern betrachtet werden. Bereits Ende des selben Jahres wurde FutureWave von Macromedia aufgekauft und das Animations-Programm in Flash umbenannt. Seitdem wurde Flash stetig weiterentwickelt und verbessert und ist schließlich zu einem beliebten Animationswerkzeug geworden[4]. Im April 2005 wurde Macromedia von Adobe aufgekauft und der Großteil des Software-Portfolios von Macromedia (darunter Flash) sind mit der Übernahme in die Adobe Creative Suite integriert worden.

Ab Flash-Version 4 war es möglich, auch komplexe Websites mit Flash zu erstellen oder aber auch ganze browserbasierte Anwendungen zu programmieren. Bis heute wurde Flash – mittlerweile in Version 10 – zum Standard für Intros auf Webseiten, animierten Werbebannern oder auch für kleine browserbasierte Online-Spiele. Doch obwohl es mit Flash schon möglich ist, sogenannte Rich Internet Applications zu erstellen, wird es von vielen Web-Entwicklern ignoriert. Vor allem wegen der „programmierunfreundlichen“ Authoring-Software galt Flash lediglich als Werkzeug für Web-Designer und Grafiker.

Seit 2004 versucht nun Adobe bzw. vormals Macromedia mit dem auf Flash basierenden Flex-Framework eine neue Plattform zu etablieren, die auch Softwareentwickler ansprechen soll. Anders wie Flash gleicht Flex eher einer Softwareentwicklungsumgebung. Das Framework umfasst neben dem als OpenSource vorliegenden Software-Development-Kit (kurz SDK), das die Flex-Klassenbibliothek und den Flex-Compiler enthält, auch die Entwicklungsumgebung Flex Builder, die Flex-Serverkomponente LiveCycle Data Service und die Flex Charting Komponenten. Mit diesen „Werkzeugen“ lassen sich laut Adobe in kürzester Zeit hoch komplexe Rich-Internet-Applications erstellen.

1.2 Ziele

Das Ziel der vorliegenden Diplomarbeit besteht u. a. darin, das Software-Framework Flex von Adobe hinsichtlich seiner Fähigkeiten zur Erstellung von Rich Internet Applications

3 vgl. The flash history, http://www.flashmagazine.com/news/detail/the_flash_history/, aufgerufen am 8. März 2008

4 vgl. The History of Flash, http://www.adobe.com/macromedia/events/john_gay/, aufgerufen am 8. März 2008

zu untersuchen und zu bewerten. Dabei soll speziell auf das Flex-Framework in Version 3 eingegangen werden.

Für die Evaluierung des Frameworks ist es notwendig, bereits etablierte Rich-Internet-Application-Frameworks vorzustellen. Ausgehend von diesen Betrachtungen sollen die Eigenheiten von Flex 3 mit Hilfe theoretischer Untersuchungen erklärt und analysiert werden. Als Ergebnis dieser Untersuchungen soll es möglich sein, eine Aussage über Anwendungsfälle treffen zu können, für die das Framework besonders geeignet bzw. ungeeignet ist.

In einem praktischen Teil dieser Arbeit soll eine Flex-Anwendung für die Firma PLUSPOL interactive GbR erstellt werden, mit der es möglich sein soll, statistische Daten von Werbespielen zu betrachten und auszuwerten. Die Programmierung der Anwendung soll zeigen, in wieweit sich Flex 3 für die Umsetzung der gegebenen Software-Anforderungen eignet.

2 Grundlagen

2.1 Framework

Ein *Framework* (engl.: Gerüst, Rahmen) bezeichnet eine konzeptionelle Struktur, die bei der Lösung von weitaus größeren Probleme oder Aufgaben helfen soll. Ein *Software-Framework* ist ein spezieller Framework-Typ, der ein halbfertiges, wiederverwendbares „Programmiergerüst“ darstellt, das die Programmierung einer Software-Anwendung erleichtert und unterstützt[1]. Der Begriff Framework gehört zu den am häufigsten benutzten Schlagwörtern in der Software-Entwicklung. *Software-Frameworks* beinhalten oft zusätzliche Programme, die die Programmierung beschleunigen sollen. Aber auch Quellcode-Bibliotheken oder spezielle Script-Sprachen werden vielen Frameworks hinzu gelegt, um die verschiedenen Software-Komponenten eines Projektes zu verbinden. Auf die verschiedenen Teile eines Frameworks wird in der Regel über APIs (Application Programming Interface; englisch für Programmierschnittstelle) zugegriffen.

Software-Frameworks lassen sich hinsichtlich ihrer Eigenschaften und Einsatzgebiete unterteilen. Einige spezielle *Software-Frameworks* werden in den nächsten Punkten kurz erläutert.

2.1.1 Application-Frameworks

Application-Frameworks dienen als Programmiergerüst für eine bestimmte Klasse von Programmen. Die Standard-Strukturen, die für diese Klassen von Anwendungen benötigt werden, sind im Framework schon implementiert. Meist wird die Implementierung des *Application-Frameworks* mittels objektorientierter Programmierung vorgenommen[2]. Das erleichtert die Umsetzung der speziellen Funktionalitäten der Anwendung, da man von bereits bestehenden Klassen erbt oder überschreibt. Mittels Application Frameworks ist es auch möglich, die gleiche Anwendung auf Basis des selben Quellcodes für verschiedene Betriebssysteme zu compilieren.

1 vgl. Richtlinien für das Framework-design, S. 25
2 vgl. Vorgehensweise zur systematischen Entwicklung datenorientierter Programmsysteme; Algorithmen und Datenstrukturen, S. 505

2.1.2 Persistence-Frameworks

Persistence-Frameworks[3] werden dazu genutzt, Daten in nicht-flüchtigen Speichermedien, wie zum Beispiel Datenbanken, zu speichern und zu laden. Das *Persistence-Framework* vereinfacht den Zugriff auf den Datenträger und ordnet die Daten des Datenspeichers auf die Datenobjekte der Applikation zu[4]. Hibernate gehört zu den populärsten *Persistence-Frameworks* für Java[5]. Es unterstützt die sogenannte Hibernate Query Language (HQL), mit der SQL-ähnliche Datenbankanfragen gestellt werden können. Der Vorteil von Hibernate besteht darin, dass die Applikation auf verschiedene Datenbanksysteme zugreifen kann, ohne Veränderungen an den Datenbankanfragen durchführen zu müssen.

2.1.3 Logging-Frameworks

Logging-Frameworks werden genutzt, um den Ablauf eines Programms zu überwachen. Vor allem bei größeren Applikationen kommen *Logging-Frameworks* zum Einsatz, da es mittels normaler Debugger meist sehr schwierig ist, Fehler im Quellcode zu finden[6]. Oft werden Logging-Informationen im Quellcode fest integriert, so dass es möglich ist, eine Applikation auch während des laufenden Betriebs zu debuggen und zu überwachen.

2.1.4 Unit-Testing-Frameworks

Unit-Testing-Frameworks werden zur Verifikation der Korrektheit von Modulen oder Klassen einer Software eingesetzt. Diese Frameworks helfen in erster Linie bei der Refakturierung von Programm-Quelltexten. Mittels *Unit-Testing-Frameworks* können alle Testfälle eines Moduls vor und nach der Code-Umstrukturierung durchlaufen und deren Ergebnisse verglichen werden[7].

2.2 Client-Server-Architektur

Die *Client-Server-Architektur* bezeichnet eine Programmarchitektur, mit der die Datenverarbeitung einer Anwendung in zwei separate Teile aufgespalten wird. Man spricht dabei auch von *kooperativer Informationsverarbeitung*. Ein Teil des Systems läuft auf dem Server und bildet die *Backend-Komponente*. Der zweite Teil arbeitet auf einer oder

3 *Persistent:* lat. „anhaltend".
4 vgl. Objektorientierung in Datenbanken; Taschenbuch Datenbanken, S. 342
5 vgl. Was ist Hibernate?; Spring & Hibernate, S. 6
6 vgl. Application Logging; Patterns for Performance and Operability, S. 81
7 vgl. Unit Testing; Applied Software Project Management, S. 156

auch mehreren Workstations und wird auch Client oder Frontend genannt[8]. Beide Teile sind über ein Netzwerk verbunden und kommunizieren mittels sogenannter *Transaktionen*. Eine Transaktion beschreibt eine Folge logisch zusammengehöriger Aktionen, beispielsweise zur Verarbeitung eines Geschäftsvorfalls[9].

Die Aufgabe des Servers (engl.: Dienstleister, Diener) besteht darin, festgelegte Dienste (Services) anderen Rechnern, den Clients, zur Verfügung zu stellen. Der Server muss dazu ständig in Bereitschaft stehen, um jederzeit auf Anfragen von Client-Rechnern reagieren zu können. Die typischen Aufgaben eines Servers sind Datenverwaltung, Berechnungen oder Druckerdienste.

Der Client hat die Möglichkeit die Dienste vom Server zu fordern, indem er sogenannte *Server-Requests* verschickt. Der Server führt anschließend den gewünschten Dienst aus und sendet einen Response (engl.: Antwort) an den Client zurück. Die Spezifikation der Kommunikation zwischen Server und Client wird – entsprechend dem jeweiligen Dienst – durch ein Protokoll festgelegt.

Die Grundidee der Client-Server-Architektur ist, die Ressourcen aller beteiligten Teile optimal ausnutzen zu können. Der bedeutende Vorteil dieser Architektur besteht darin, dass das Gesamtsystem jegliche Größenordnung annehmen kann. Dabei ist die verhältnismäßige Leistung der Architektur-Teile nicht vorgegeben, d. h. das Leistungsvermögen des Clients kann durchaus das des Servers übersteigen. Die Dimensionierung von Client und Server hängt allein vom Einsatz des Gesamtsystems ab.

2.3 Webservices

Das World Wide Web Consortium (W3C) bezeichnet Webservices[10] als Dienste, die eine interoperable Kommunikation zwischen Computern über ein Netzwerk ermöglichen. Die Definition eines Webservice ist sehr weitläufig und umfasst eine große Anzahl verschiedenster Architekturen. Möchte man Webservices sehr allgemeingültig beschreiben, so beinhaltet die Definition zumindest die Begriffe Client und Server.

Als Webservices werden generell Programm-APIs genannt, die über ein Netzwerk, wie zum Beispiel das Internet, erreichbar sind. Das Ziel der Webservice-Technologie liegt darin, verschiedene Softwaresysteme unabhängig von Plattform und Programmiersprache miteinander kommunizieren und arbeiten zu lassen. Der Datenaustausch innerhalb dieses

8 vgl. Zweischichtige Client-Server-Architektur; Taschenbuch Rechnernetze und Internet, S. 139
9 vgl. Client-Server-Architektur, http://gd.tuwien.ac.at/study/hrh-glossar/1-2_17.htm, aufgerufen am 8. März 2008
10 vgl. Web Services Activity, W3C: http://www.w3.org/2002/ws/, aufgerufen am 8. März 2008

Systems funktioniert mittels XML-basierter Nachrichten, deren Übertragung auf beliebigen Transportprotokollen basieren kann. Für diese Übertragung der Daten zwischen den Systemen hat sich das XML-basierte Simple Object Access Protocol (SOAP) etabliert. Das SOAP-Protokoll stellt Konventionen auf, wie die übertragenen Informationen abzubilden und zu interpretieren sind[11].

Ein üblicher Aufbau eines Webservices besteht aus einem Server, der seine Dienste (Services) anderen Systemen über ein Netzwerk zur Verfügung stellt, und einem Client, der diese Dienste beanspruchen kann. Doch die Webservice-Architektur sieht noch einen dritten optionalen Teil vor: den *Service-Broker*. Der Service-Broker stellt dem System eine Beschreibung der vorhandenen Services des Servers zur Verfügung. Diese Informationen stehen in Form eines Dokuments zur Verfügung, das mit der Webservice Description Language (WSDL) verfasst wurde. Ein Service-Broker ist für einen funktionierenden Webservice nicht zwingend notwendig, aber er ist Voraussetzung für eine automatisierte Codegenerierung auf der Client-Seite.

Um die Interoperabilität von Webservices zu verbessern, hat die WS-I (Web Services Interoperability Organization) sogenannte Profile veröffentlicht. Ein Profil beinhaltet die Spezifikationen einer funktionierenden Konfiguration eines Webservices. Diese Beschreibung umfasst unter anderem die Art der Kommunikation zwischen Server und Client (z. B. SOAP), die Version der Webservice-Beschreibungssprache (z. B. WSDL 2.0) und zusätzliche Eigenschaften, die von den Standard-Spezifikationen eines Webservices abweichen.

Adobe veröffentlichte mit Flash 6 ein eigenes Format zur Übertragung der Daten zwischen Server und Client. Dieses sogenannte Action Messaging Format (AMF) überträgt Daten in Binärform und basiert in den Grundzügen auf dem SOAP-Standard[12]. AMF wurde speziell für den Datenaustausch zwischen Flash-Applikationen (Client) und einer Datenbank bzw. Server entwickelt. Die binären Daten einer AMF-Message können von Flash direkt in ActionScript-Objekte umgewandelt werden. Damit entfällt ein aufwendiges Parsen der übertragenen Informationen, wie es zum Beispiel bei XML-basierten Daten notwendig wäre. Da mit Adobe Flash 9 zahlreiche neue Datentypen und Features hinzugekommen sind, wurde das bisherige AMF 0 durch AMF 3[13] abgelöst.

2.4 Rich-Internet-Application

Aufgrund der Server-Client-Architektur von herkömmlichen Internetseiten sind bei Benutzeraktivitäten mit dem Seiteninterface immer mit kleinen Wartezeiten und

11 vgl. Definition Webservices; Collaborative Business und Web Services, S. 59
12 vgl. Why use Flash Remoting?; Flash Remoting, S. 348
13 vgl. Spezifikationen von AMF 3,
http://download.macromedia.com/pub/labs/amf/amf0_spec_121207.pdf, aufgerufen am 8. März 2008

Verzögerungen zu rechnen, bis sich eine neue Seite aufgebaut hat. Rich Internet Applications (RIAs) sind Webanwendungen, die nicht wie traditionelle Internetseiten auf Benutzerinteraktionen reagieren, sondern sich wie Desktopanwendungen verhalten[14]. Dem Nutzer stehen dabei eine große Anzahl an Interaktionsmöglichkeiten und -komponenten, wie zum Beispiel Schaltflächen, Menüs oder Fensterdialoge zur Verfügung, wie man sie seit Jahren aus kommerziellen Softwareprodukten kennt. Aus diesen Gründen entsteht ein Nutzungsgefühl, das als „reichhaltig“ (engl.: rich) bezeichnet werden kann.

RIAs laufen für gewöhnlich in Internet-Browsern. Eine Installation ist nicht nötig, da die Applikation automatisch beim Aufruf der Seite geladen wird. Jeder Besucher der Website nutzt somit automatisch immer die neueste Version des Programms. Dies macht den Einsatz von RIAs vor allem im Intranet interessant, da regelmäßige aufwendige Updates der firmeninternen Software entfallen. Außerdem ermöglicht die Betrachtung der RIAs in Browsern einen plattformunabhängigen Betrieb.

Der Funktionsumfang und die Komplexität einer RIA hängt ausschließlich von der Performance des Client-Rechners ab. In der Regel werden RIAs so entwickelt, dass die Reaktionszeit des Benutzerinterfaces auf ein Minimum reduziert wird und möglichst viele Berechnungen, für die sonst der Server zuständig ist, vom Client-PC durchgeführt werden. Dennoch existiert weiterhin eine synchrone Kommunikation zwischen Browser und Server. Doch RIAs bieten zusätzlich noch eine asynchrone Kommunikation, die im Hintergrund abläuft und dem Nutzer verborgen bleibt. Das Prinzip der asynchronen Kommunikation kommt besonders beim sogenannten Prefetching zum Einsatz. Bei dieser Technik versucht das Programm zu ermitteln, welche neuen Daten der Nutzer demnächst benötigen wird und lädt diese im Voraus. Bekanntestes Beispiel für den Einsatz dieser Technik ist Google Maps[15]. Hier werden die benachbarten Kartenteile des aktuellen Sichtbereichs geladen, bevor der Nutzer die Karte verschiebt.

Rich-Internet-Applications laufen in einer sicheren lokalen Umgebung, die als Sandbox bezeichnet wird. Das Programm erhält somit aus Sicherheitsgründen nur einen eingeschränkten Zugang zum System des Clients. Die Sandbox selbst muss also für einen reibungslosen Betrieb einer RIA auf dem Client installiert sein.

Einen weiterer Nachteil stellt sich für Webdesigner, die ihre Internet-Applikation gern suchmaschinenfreundlich gestalten möchten. So ist es bei den meisten RIA-Technologien nicht ohne Weiteres möglich eine reibungslose Indizierung der textuellen Inhalte der Anwendungen durch Suchmaschinen zu gewährleisten. Auch von einem barrierefreien Zugang der Inhalte kann in den meisten Fällen nicht gesprochen werden. Screenreader können beispielsweise die

14 vgl. Rich Internet Applications (RIAs); AJAX, Rich Internet Applications, and Web Development for Programmers, S. 32

15 Google Maps, http://maps.google.de/, aufgerufen am 8. März 2008

Inhalte vieler RIA-Technologien nur schwer oder gar nicht auslesen.

Im folgenden Kapitel 3 werden einige der bekanntesten Rich-Internet-Application-Frameworks vorgestellt.

3 Rich-Internet-Application-Frameworks

Neben Adobe Flex 3, das in Kapitel 4 näher betrachtet wird, existieren noch zahlreiche weitere RIA-Frameworks. Welche Technologie für den Entwickler und für den Nutzer optimal ist, hängt nicht zuletzt vom Einsatzprofil der Anwendung ab. Die vorgestellten Frameworks sind, beginnend mit den bereits 1995 entwickelten Java-Applets, hinsichtlich ihres Veröffentlichungsdatums geordnet.

3.1 Java-Applet

Die Java-Applets-Technologie[1] wurde bereits 1995 – zusammen mit der ersten Version der Programmiersprache Java – von Sun veröffentlicht und stellt somit die erste RIA-Plattform dar. Ein Java-Applet besteht aus Java-Bytecode, d. h. der Programmquellcode muss zuvor mit einem Java-Compiler übersetzt werden. Für gewöhnlich werden Applets in der Programmiersprache Java geschrieben, doch auch andere Programmiersprachen können verwendet werden – vorausgesetzt der Programmcode kann in Java-Bytecode compiliert werden.

Ein Java-Applet kann entweder in einem Web-Browser oder in Suns Applet-Viewer – meist für Testzwecke – ausgeführt werden. In beiden Fällen laufen die Programme, wie auch die meisten anderen RIA-Technologien, in „abgeschotteten“ Laufzeitumgebungen (Sandbox), so dass kein unkontrollierter Zugriff auf das lokale Dateisystem möglich ist. Um in Browsern Java-Applets nutzen zu können, muss entweder die Java-VM bereits im Browser integriert sein (Microsoft IE bis Version 5, Netscape 3.x und 4.x) oder als Plugin nachinstalliert werden. Die Java-VM ist für die meisten Betriebssysteme, einschließlich Windows, Unix, Mac OS und Linux erhältlich. Auch moderne Browser, wie der Mozilla Firefox oder Opera werden unterstützt. Neben der kostenlosen, aber nicht offenen Java-VM vom Java-Hersteller Sun existieren noch andere OpenSource VMs wie beispielsweise *Kaffe*. Vorteil von Kaffe gegenüber der JVM von Sun ist, dass sie deutlich kleiner und auf über 50 Betriebssystemen lauffähig ist.

Dank der langjährigen kontinuierlichen Weiterentwicklung bietet Java eine große Funktionsvielfalt an fertigen Paketen und Bibliotheken. Java-Applets bieten dabei den

1 vgl. Applets, Sun Developer Network, http://java.sun.com/applets/, aufgerufen am 8. März 2008

kompletten Funktionsumfang der J2SE-API (Java 2 Standard Edition API) in der aktuellen Version 6. In Verbindung mit Servlets (kleine Applikationen auf einem Java-Webserver) oder Java Application Servern sind somit komplexe Anwendungen erstellbar. Des Weiteren erleichtern eine sehr große Nutzergemeinschaft mit unzähligen Foren, Tutorialseiten und Beispielanwendungen die Entwicklung von Java Applets.

Die Nachteile von Java-Applets konzentrieren sich hauptsächlich auf deren Laufzeit-Umgebung. So kommt zu der Zeit, die für das vollständige Laden und Initialisieren der Anwendung benötigt wird, noch die vergleichsweise lange Initialisierungszeit der über 12 MB großen Java-VM hinzu. Aus diesen Gründen sind größere Java Applets eher selten auf Websites im Internet anzutreffen. Einen weiteren Nachteil besteht in der Tatsache, dass die Inhalte von Java Applets nicht von Suchmaschinen erfasst werden können. Aus diesen Gründen werden Java Applets häufig in Firmen-Intranets eingesetzt.

3.2 Adobe Flash

Flash ist eine proprietäre integrierte Entwicklungsumgebung und wurde 1996 von Macromedia vorgestellt. Nach der Übernahme von Macromedia wird Flash von Adobe konsequent voran getrieben und hat sich zu einem sehr beliebten Werkzeug für Vektor-Animationen und interaktive Anwendungen entwickelt[2]. Für Werbebanner und -animationen hat sich Flash sogar zum weltweiten Standard etabliert[3]. Auch für kleine, browserbasierte Spiele erfreut sich Flash größter Beliebtheit. Viele Webdesigner nutzen kleine Flash-Animationen als Intro für ihre Internetseiten. Aber auch große Unternehmen, wie der Web-Video-Gigant YouTube[4], der einen mit Flash erstellten Videoplayer nutzt, setzt auf diese Technologie.

Der Kern des Flash-Frameworks besteht aus dem Flash Authoring-Tool. Das Programm ist ein leistungsstarkes Design- und Programmierwerkzeug mit dem multimediale Inhalte erstellt werden können. Mit Flash lassen sich nicht nur einfache Vektor-Grafiken erstellen sondern auch animieren. Zusätzlich können Bitmaps und sogar ganze Videos eingebettet werden. In Verbindung mit der Flash-Programmiersprache ActionScript können auch interaktive Anwendungen programmiert werden.

Das Flash-Authoring-Tool kompiliert die erstellten Quelldateien zu Flash-Bytecode und speichert sie als „swf“-Dateien (Shockwave Flash) ab. Um diese Dateien betrachten bzw. abspielen zu können, wird der Adobe Flash Player als Sandbox benötigt. Der Flash Player ist im Gegensatz zur Flash-IDE kostenlos und kann von der Herstellerwebsite für verschiedenste

2 vgl. RIA Technologies; Adobe Flex 3: Training from the Source, S. 10
3 vgl. Flash-Banner; TYPO3 Online-Marketing-Guide: Affiliate- und E-Mail-Marketing, Keyword-Advertising, Suchmaschinen-Optimierung mit TYPO3, S. 44
4 siehe YouTube, www.youtube.de

Betriebssysteme und Browser in der aktuellen Version heruntergeladen werden. Zur Integration einer swf-Datei auf einer Website reichen schließlich nur wenige Zeilen HTML-Code aus.

Des Weiteren besteht das Flash-Framework aus den serverseitigen Technologien Flash-Remoting und Flash Media Server. Flash-Remoting ist eine Technik zur effektiven und effizienten Client-Server-Kommunikation. Dabei arbeitet Flash-Remoting ähnlich wie Webservices, d. h. es ermöglicht das Ausführen von Service-Methoden des Servers. Diese Methoden können in PHP, Java, .NET oder jeder anderen Programmiersprache geschrieben werden. Zur Übertragung der Daten wird das binäre Übertragungsformat AMF (Action Message Format) genutzt, das vom Adobe Flash Player nativ unterstützt wird und gegenüber XML-basierten Webservices eine signifikant schnellere Datenübertragung ermöglicht. Um eine Flash-Remoting-Kommunikation aufbauen zu können wird ein serverseitiger Gateway benötigt, der fähig ist, ankommende AMF-Anfragen den zugehörigen Services zuzuordnen und die entsprechende Antwortdaten zurückzusenden. Adobe liefert dazu in seinen Produkten ColdFusion und JRun (J2EE Application-Server) einen Remoting-Server aus. Auch eine standalone Version des Remoting-Servers ist von Adobe erwerbbar, die für J2EE Application-Servers oder .NET-Servern ausgelegt ist. Weiterhin existieren zahlreiche freie Entwicklungen wie WebOrb[5], AMFPHP[6] oder OpenAMF[7].

Mit dem Flash Media Server bietet Adobe eine leistungsstarke Plattform für Video-on-Demand[8] und Streaming-Video[9] an. Die noch bis zur Version 2.0 unter dem Namen Flash-Communications-Server bekannte Technologie unterstützt unter anderem Live-Video, um eigens mit der Webcam aufgenommenes Filmmaterial anderen zur Verfügung zu stellen oder auf dem Server zu speichern[10]. Weiterhin bietet die Technologie Echtzeit-Kommunikation zwischen mehreren Clients, wie sie zum Beispiel für Live-Chats oder Multiplayerspiele benötigt wird.

3.3 OpenLaszlo

OpenLaszlo ist ein Entwicklungsframework für Rich Internet Applications und ist auch unter dem Namen Laszlo Presentation Server (LPS) bekannt. Das kalifornische Softwareunternehmen Laszlo Systems begann 2001 mit der Entwicklung von OpenLaszlo und

5 siehe WebOrb Website, http://www.themidnightcoders.com/products/weborb-for-php
6 siehe AMFPHP Website, http://www.amfphp.org/
7 siehe OpenAMF Website, http://sourceforge.net/projects/openamf/
8 Video-on-Demand: beschreibt die Möglichkeit digitales Videomaterial auf Anfrage von einem Internetdienst herunterzuladen.
9 Streaming-Video: kontinuierliche Übertragung von Videodaten, die das Betrachten eines Videos ermöglicht, bevor die Datei vollständig heruntergeladen ist.
10 vgl. Recording and Playing Back Streaming Audio and Video; Learning Flash Media Server 2, S. 18

veröffentlichte bereits Anfang 2002 die erste Version. Im Oktober 2004 stellte der Entwickler die kompletten Sourcen des Frameworks unter die Common Public License (GPL) und gründete das OpenLaszlo Projekt. Mit dem Release von Version 3 wurde schließlich die Bezeichnung des Laszlo Presentation Servers in OpenLaszlo umbenannt. Die neueste Version 4.2 kann auf der Website des OpenLaszlo-Projektes[11] kostenlos heruntergeladen werden.

OpenLaszlo-Programme werden mit der vollständig objektorientierten LZX-Programmiersprache geschrieben, die eine Kombination aus XML und JavaScript darstellt. Die Programm-Sourcen werden mit dem OpenLaszlo-Server (Java Server Applet) kompiliert und als SWF-Datei ausgegeben. Ab OpenLaszlo 4 ist neben Flash 7 und 8 sogar eine Kompilierung in DHTML möglich, ohne eine Änderung am Sourcecode durchführen zu müssen. Damit verfolgt das Laszlo Projekt das Ziel, den „Player“ der Anwendung herstellerunabhängig zu machen. So sollen in den nächsten OpenLaszlo-Versionen weitere Exportmöglichkeiten folgen.

Einen weiteren Schritt in Richtung Plattformunabhängigkeit vollzog das OpenLaszlo-Projekt, indem es eine Kooperation mit Sun Microsystems einging – dem Entwickler der Java Technologie. Ziel dieser Zusammenarbeit unter dem Namen „Orbit“ ist es, OpenLaszlo-Anwendungen auch auf Geräten und Systemen lauffähig zu machen, die auf der Java-Plattform basieren. Speziell auf mobilen Geräten, die die Java Micro Edition (Java ME) unterstützen, sollen in Zukunft Rich Internet Applications mit Hilfe von OpenLaszlo entwickelt werden können.

Zum Programmieren einer OpenLaszlo-Anwendung, also zum Erstellen der LZX-Dateien, ist lediglich ein Texteditor nötig. Im Rahmen des OpenSource Projektes wurde die kostenlose Entwicklungsumgebung IDE4Laszlo veröffentlicht, die das Editieren erheblich erleichtern soll. Die IDE basiert auf Eclipse und kann von der Projekt-Website heruntergeladen werden. Neben IDE4Laszlo existieren noch OpenSource-Projekte und kommerzielle Produkte.

Die OpenLaszlo Entwickler haben zwei Varianten vorgesehen eine OpenLaszlo Anwendung einzusetzen (Deployment)[12].

SOLO Deployment

Bei einem SOLO Deployment (Standalone OpenLaszlo Output) wird die vorkompilierte Anwendung auf einem HTTP-Server zur Verfügung gestellt. Diese Variante unterstützt eine Anbindung zu statischen Datenquellen wie XML- oder Text-Dateien, aber auch zu dynamischen Inhalten via Webservices oder einfachen POST- oder GET-Server-Anfragen. Der Vorteil eines SOLO Deployments besteht darin, dass es vergleichsweise geringe Anforderung

11 vgl. OpenLaszlo, http://www.openlaszlo.org/, aufgerufen am 4. Februar 2009
12 vgl. Deployment Architecture, OpenLaszlo, http://www.openlaszlo.org/deparchitecture, aufgerufen am 4. Februar 2009

an den Server stellt.

OpenLaszlo Server

Mit einem OpenLaszlo Server Deployment ist es möglich eine Applikation innerhalb einer J2EE oder Java-Servlet-Umgebung zur Laufzeit zu kompilieren. Bei diesem Deployment stehen alle Datenanbindungen eines SOLO Deployments zur Verfügung. Zusätzlich dazu kommen noch Datenintegration via XML-RPC oder Java-RPC. Laut Herstellerseite bietet ein OpenLaszlo Server die Skalierbarkeit und Zuverlässigkeit, wie sie für RIA-Anwendungen und Websites mit hohen Ansprüchen benötigt werden.

3.4 Ajax

Ajax ist ein Akronym und steht für die Wortfolge „Asynchronous JavaScript and XML“. Das Akronym beschreibt eine *asynchrone* Datenübertragung im *XML*-Format zwischen Server und Client. Der Datenfluss auf der Seite des Client wird dabei mittels *JavaScript* gesteuert.

Die technologische Grundlage von Ajax wird generell mit dem Begriff *XMLHttpRequest* beschrieben. XMLHttpRequest ist eine JavaScript-API, die zum Transfer von beliebigen Daten über das HTTP-Protokoll verwendet werden kann. Websites, die auf dieser Technologie basieren, haben den Vorteil, dass sie HTTP-Anfragen durchführen können, ohne die komplette Seite neu zu laden. Das heißt, es werden nur bei Bedarf Daten nachgeladen und die gesamte Web-Anwendung reagiert somit schneller auf Benutzereingaben. Das Grundprinzip und auch die damit verbundenen Technologien existieren schon seit etwa 1998. Zu dieser Zeit entwickelte Microsoft die sogenannte Remote-Scripting-Komponente, die das clientseitige Auslösen von HTTP-Anforderungen ermöglichte. Die Komponente wurde anschließend durch das Outlook-Web-Access-Team verfeinert, das an einem webbasierten E-Mail-Client arbeitete. In Folge dessen bekam die Komponente XML-Unterstützung und wurde fester Bestandteil des Internet Explorers 4.0. Zwar basiert Outlook Web Access nicht auf dem XMLHttpRequest-Objekt, doch stufen Experten diese Anwendung als ersten erfolgreichen Vertreter des Ajax-Konzeptes ein. Die Herkunft des Begriffes „Ajax“ selbst ist nicht mehr eindeutig nachvollziehbar. Die erste Verwendung des Begriffs ist zumindest durch einen Aufsatz[13] von Jesse James Garrett[14] vom 18. Februar 2005 belegt. Zwar waren zu diesem Zeitpunkt die technologischen Grundlagen unter dem Begriff XMLHttpRequest schon bekannt, doch zumindest wurden mit diesem Aufsatz Software-Technologien und -Konzepte nun unter dem Namen Ajax zusammengefasst.

13 vgl. Ajax: A New Approach to Web Applications, http://www.adaptivepath.com/ideas/essays/archives/000385.php, aufgerufen am 1. März 2009

14 Mitarbeiter der Agentur Adaptive Path

Die Werkzeuge zum Erstellen einer Ajax-Website im Vergleich zu „herkömmlichen“ Internetseiten unterscheiden sich kaum. Um die visuelle Struktur zu konstruieren, kommt statt HTML dessen Erweiterung XHTML zum Einsatz. Dabei wird beim Seitenaufbau strikt nach dem Document Object Model (DOM) vorgegangen. Mittels JavaScript ist es möglich, das DOM zu manipulieren, wodurch eine dynamische Darstellung der Inhalte möglich wird.

Die vom Server angeforderten Daten können in beliebiger Form – häufig im XML-Format – empfangen werden. Die Datenübertragung bei Ajax basiert auf der Client-Server-Architektur. Allerdings wird auf beiden Seiten – also beim Client und beim Server – eine entsprechende Ajax-Komponente benötigt.

Client-Seite

Auf Client-Seite wird ein Browser mit XMLHttpRequest-Unterstützung und aktiviertem JavaScript benötigt. Jegliche Ajax-Funktionalität und der dynamische Seitenaufbau wird mit JavaScript realisiert.

Server-Seite

Auf dem Server wird die eigentliche Programmlogik der Anwendung installiert. Mit welcher spezifischen Technik die Logik umgesetzt werden soll, sieht das Ajax-Konzept allerdings nicht vor. So existieren zahlreiche serverseitige Ajax-Lösungen, wie Enterprise Java Beans und .NET-Komponenten oder auch Skript-Komponenten, wie zum Beispiel die PHP-Frameworks AjaxAC und Cajax.

Vor- und Nachteile von Ajax

Der entscheidende Vorteil von Ajax gegenüber anderen RIA-Frameworks besteht darin, dass kein spezielles Browserplugin und keine virtuelle Maschine zum Starten einer Ajax-Anwendung benötigt wird. Des Weiteren ermöglicht die Unterstützung des MVC-Architekturmusters die Entwicklung sehr komplexer aber trotzdem noch gut erweiterbarer Anwendungen.

Da Ajax komplett auf JavaScript basiert und keine eigene Laufzeitumgebung benötigt, ergeben sich auch einige Nachteile. Ist beispielsweise kein JavaScript im Browser aktiviert, kann die Anwendung gar nicht erst starten. Auch das dynamische Nachladen der Daten kann Probleme verursachen. So ist die Zurück-Funktion und das Speichern von Lesezeichen des Browsers ohne spezielle Anpassungen der Ajax-Anwendung nicht möglich.

3.5 Microsoft Silverlight

Silverlight bezeichnet eine Web-Präsentationstechnik des amerikanischen Softwareunternehmens Microsoft. Die Technologie stellt vektorbasierte Inhalte auf Basis der

Windows Presentation Foundation dar. Zum ersten Mal wurde Silverlight am 16. April 2007 auf der NAB2007-Konferenz (National Association of Broadcasters) in Las Vegas vorgestellt und ist die jüngste hier vorgestellte Technologie.

Bis zu seiner offiziellen Vorstellung war Silverlight auch unter dem Namen WPF/E (Windows Presentation Foundation/Everywhere). Dabei handelt es sich um eine stark „abgespeckte" Version der WPF, der Grafik API von Microsoft Windows. WPF ist ein Framework zur Ausgabe von 2D/3D-Vektor-Grafiken, Video und Audio sowie Bildern. Auch Animationen sind mit WPF möglich. Silverlight besticht vor allem durch seine leistungsstarke Videowiedergabe bis zur HDTV-Auflösung 720p und 3D-Animationen, die bislang in graphischen Benutzeroberflächen nur selten zu sehen waren. Dank kompletter Hardwareunterstützung kann Silverlight auch die Ressourcen von Grafikkarten nutzen und somit die Auslastung des Gesamtsystem schonen. Microsoft bietet das Browser-Plugin zum Betrachten von Silverlight-Anwendungen für Windows und Mac in der Version 2 kostenlos zum Download[15] an. Mit dem Ziel auch eine Linuxunterstützung des Silverlight-Players zu entwickeln, wurde das OpenSource Projekt Mono ins Leben gerufen. Zum Zeitpunkt der Erstellung dieser Arbeit ist die Linux-Version des Players unter dem Namen Moonlight in Version 2.2 erhältlich[16]. Auch für mobile Geräte, die Windows Mobile 6 unterstützen, wird derzeit an einer Laufzeitumgebung für Silverlight gearbeitet.

Silverlight-Anwendungen werden mit XAML geschrieben, der „eXtensible Application Markup Language". Das XAML-Format wurde von Microsoft entwickelt und ist im Grunde dem XML-Format sehr ähnlich. Mit dieser „Beschreibungssprache" werden die Komponenten der Anwendung definiert, wie z. B. Buttons, Textfelder, Videos oder auch einfache geometrische Objekte wie Linien, Dreiecke und Vierecke. XAML stellt außerdem eine Reihe vordefinierter Animationsabläufe, die optional noch miteinander kombiniert werden können. Mittels .NET-Scriptsprachen ist auch eine programmatische Manipulation der Benutzeroberfläche möglich.

Microsoft bietet zwei Programme zum Entwickeln von Silverlight-Anwendungen an: das neue Visual Studio mit Codenamen „Orcas" und Microsoft Expression Blend.

Microsoft Visual Studio

Microsoft Visual Studio dient zum Erstellen, Kompilieren und Debuggen von Silverlight-Anwendungen. Dem Programmierer steht dabei frei, mit welcher Standard-.NET-Sprache (z. B.: JavaScript, Ruby, Python, C# oder Visual Basic .NET) er seine Anwendung erstellen möchte. Ab der Version 2008 von Visual Studio sind alle wichtigen

15 Silverlight Download, http://www.microsoft.com/silverlight/resources/install.aspx, aufgerufen am 2. Februar 2009

16 Silverlight Mono 2.2 Download, http://www.go-mono.com/mono-downloads/, aufgerufen am 2. Februar 2009

Bibliotheken und Klassen enthalten, die zur Programmierung von Silverlight-Anwendungen nötig sind. Damit verbindet Microsoft die von Programmierern bekannte Benutzeroberfläche von Visual Studio mit der neuen Technologie, um somit einen möglichst leichten Um- bzw. Einstieg in Silverlight zu ermöglichen.

Microsoft Expression Blend

Expression Blend gehört zur Microsoft Expression-Familie und ist an Designer gerichtet, die ohne Programmieraufwand Silverlight-Oberflächen gestalten möchten. Dabei steht ihnen eine Vielzahl an Gestaltungsmöglichkeiten zu Verfügung, etwa Farbverläufe oder vektorbasierte Elemente. Ähnlich wie mit der Flash Authoring-Software lassen sich mit Expression Blend auch Animationen auf Basis von Zeitleisten kreieren. Das Programm wandelt alle graphischen Daten in Visual-Studio-Code um und bindet die Befehle direkt in ein bestehendes Projekt ein. Somit ist eine bessere Trennung von Darstellungs- und Steuerungsschicht des Programms möglich.

4 Adobe Flex 3

Nachdem die Grundlagen der Technologien, die rund um Rich Internet Applications existieren, in Kapitel 2 erläutert und auch einige RIA-Technologien in Kapitel 3 vorgestellt worden sind, wird nun das Adobe Flex 3-Framework genauer betrachtet. In Abschnitt 4.1 werden die Besonderheiten der Flex 3-Technologie dargestellt. In den darauf folgenden Abschnitten werden zum einen die Software-Tools vorgestellt, die im Flex-Framework integriert sind. Des Weiteren werden die Programmiersprachen MXML und ActionScript 3 sowie die im Flex 3 SDK vorhandene Quellcode-Bibliothek analysiert. Zum Ende dieses Kapitels (Abschnitt 4.9) wird ein Überblick über die vorherigen Flex-Versionen und ein Ausblick auf die zukünftige Entwicklung von Flex gegeben.

4.1 Das Framework

Den Kern des Adobe Flex Frameworks bildet eine ActionScript 3.0 Bibliothek. Sie stellt die Grundlage für die Erstellung von flashbasierten RIAs dar. Flex 3 ist im Vergleich zu Flash „programmiererorientierter“ und stellt dem Software-Entwickler eine ausgereifte Software-Architektur zur Verfügung. Vielen Programmierern aus dem .NET- oder Java-Umfeld dürfte Flex sehr entgegen kommen, da viele Design-Pattern starke Ähnlichkeiten mit diesen beiden Technologien besitzen.

Anders wie bei Flash, wo man jedes Objekt aufwendig zeichnen muss, bietet Flex ein reichhaltiges Angebot an vorgefertigten Komponenten und Effekten, wie zum Beispiel Buttons, Listen, Menüs, Slider, Tabulatoren und vieles mehr. Aber auch eigene Komponenten lassen sich leicht erstellen, indem man die vorgefertigten Komponenten nach seinen Wünschen anpasst. Flex bietet zu diesem Zweck zahlreiche Möglichkeiten zu visuellen Anpassungen bzw. Änderungen am Look-And-Feel der Komponenten.

Jegliche Steuerung des Programmflusses einer Flex-Anwendung wird mit Events geregelt. So triggert beispielsweise jede Komponente bei Statusänderung ein entsprechendes Event, das vom Programm registriert wird und darauf reagieren kann. Das Event-Model von Flash besitzt dabei große Ähnlichkeit mit der Programmiersprache Java.

Flex-Anwendungen werden für gewöhnlich mit einem Mix aus ActionScript 3 und der XML-basierten Programmiersprache MXML erstellt. Mit MXML wird die Benutzeroberfläche

einer Anwendung definiert. Jeder MXML-Tag stellt dabei eine Flex-Komponente dar. Erstellt man eigene Komponenten, sind diese als eigene Tags verfügbar. Die Eigenschaften der Komponente werden dann zu Attributen der Tags. Die Programmlogik wird in ActionScript 3 geschrieben. ActionScript 3 bietet dazu neben der Flex-Bibliothek auch alle AS-Klassen, die in Flash zur Verfügung stehen.

Für die Entwicklung einer Flex-Applikation ist grundsätzlich nur das kostenlose Flex-Framework SDK nötig. Dies beinhaltet unter anderem die vordefinierten ActionScript-Klassen und MXML-Komponenten. Doch Kernstück des SDK ist der Kommandozeilen-Compiler. Er kompiliert die Flex-Anwendung aus allen MXML-Dateien, den zugehörigen ActionScript-Klassen und den sonstigen eingebundenen Ressourcen.

4.1.1 Besonderheiten von Adobe Flex 3

Flex bietet die typischen Vorteile von Rich-Internet-Application-Frameworks. Einige der nennenswertesten Vorteile werden im Folgenden kurz erläutert.

Plattformunabhängigkeit

Flex bietet weitreichende Plattformunabhängigkeit. Voraussetzung dafür ist eine Flash-Player-Version für die jeweilige Plattform. Für Flex 3 ist der Flash Player 9 notwendig, der für die meisten Betriebssysteme und Browser angeboten wird[1]. Adobe ist dazu Kooperationen mit zahlreichen wichtigen Geschäftspartnern eingegangen, darunter Microsoft, Apple, Netscape, Novell, TurboLinux, Red Hat und AOL.

Auslagerung der Darstellung auf Client

Die komplette Visualisierung einer Flex-GUI wird auf der Client-Maschine vollzogen. Dies hat zur Folge, dass der Server keine aufwendigen graphischen Renderings durchführen muss und somit entlastet wird.

Geringe Netzwerklast

Abgesehen vom einmaligen Laden des UI werden nur Nutzdaten vom Server zum Klienten transportiert. Flex bietet außerdem Möglichkeiten CSS-Dateien, Grafiken u. ä. auszulagern, die nur bei Bedarf nachgeladen werden. Nachgeladenen Dateien – einschließlich der Flex-Anwendung selbst – können vom Browser gecached werden, so dass bei einem wiederholten Besuch der Seite kein erneutes Übertragen der Anwendung nötig ist.

1 vgl. Flex FAQ, http://www.adobe.com/de/products/flashplayer/productinfo/faq/, aufgerufen am 8. März 2009

Kostenloses SDK

Das Flex SDK steht bereits ab Version 2 kostenlos zum Download bereit[2]. Damit sinken die Entwicklungskosten von Flex-Applikationen.

Dokumentation und Support

Adobe hat für Flex eine vollständige Dokumentation zu allen Flex-Klassen online zur Verfügung gestellt[3].

4.1.2 Nachteile von Adobe Flex 3

Flex-Anwendungen unterliegen den typischen Nachteilen von Rich Internet Applications. Adobe hat versucht den meisten RIA-Problemen zu begegnen und hat einzigartige Technologien in Flex 3 integriert, die im Folgenden kurz vorgestellt werden.

Erhöhter Download-Traffic

Eine Flex-Anwendung, die auf einer Website integriert ist, muss vom Client erst vollständig vom Server heruntergeladen werden, bevor sie gestartet werden kann. In Abhängigkeit von der Internetverbindung kann es dadurch bei großen Applikationen zu langen Wartezeiten kommen, bis die ersten Inhalte auf dem Bildschirm zu sehen sind.

Flex entgegnet dem Problem zumindest teilweise, indem es dem Programmierer die Möglichkeit gibt, die Applikation modular aufzubauen. So brauchen nur die wesentlichsten Module beim Starten geladen werden. Die übrigen Teile können bei Bedarf nachgeladen werden. Doch selbst eine „leere“ SWF-Datei, die mit Flex 3 compiliert wurde, benötigt ca. 123 KByte Speicherplatz. Dieser Wert ist im Vergleich zu einem Flash-Film, der mit dem Flash-Authoring-Tool erstellt wurde und nur wenige hundert Bytes beträgt, recht groß. Für eine Rich Internet Application sind solche Größenordnungen allerdings durchaus noch im normalen Bereich. Der Grund für die erhöhte Dateigröße liegt darin, dass beim Compilieren einer Flex-Anwendung immer die Standard-Klassen des Flex-Frameworks eingebunden werden. Folgerichtig wächst die Dateigröße nur gering, wenn Standard-Komponenten des Flex-Frameworks in der Applikation genutzt werden.

Auch multimediale Inhalte, wie Bilder und Sound-Dateien, können in eine Flex-Applikation eingebettet werden. Das kann die Dateigröße des Flashfilms dramatisch vergrößern. Doch auch hier hat man Möglichkeiten verschiedenste Inhalte erst im Nachhinein zu laden.

2 vgl. Pressemitteilung von Adobe, http://www.adobe.com/aboutadobe/pressroom/pressreleases/200704/042607Flex.html, aufgerufen am 9. März 2009

3 vgl. Adobe Flex 3 Language Reference, http://livedocs.adobe.com/flex/3/langref/, aufgerufen am 9. März 2009

Bedarf einer Laufzeitumgebung

Ein weiterer Nachteil von Flex ist die Notwendigkeit eines installierten Adobe Flash Players (mindestens Version 9). Diese Laufzeitumgebung muss zwingend auf dem Client-Rechner vorhanden sein. Der Flash Player 9 kann zwar einfach nachinstalliert werden, doch bei Nutzern, die nur über wenig Rechte auf dem Client-Rechner verfügen, kann es zu Komplikationen kommen.

Geringe Suchmaschinenfreundlichkeit

Ein weiteres Problem liegt in der geringen Suchmaschinenfeundlichkeit von Flash-Filmen. Suchmaschinen sind in der Regel darauf ausgelegt, einfache Dokumente, wie HTML- oder TXT-Dateien auszulesen und zu indizieren. SWF-Dateien liegen allerdings im Byte-Format vor und sind daher für die Suchmaschine nicht lesbar.

Der Flash-Entwickler Macromedia hat zur Beseitigung dieser Barriere bereits 2003 eine Zusammenarbeit mit den führenden Suchmaschinen-Anbietern, wie Google und Yahoo! begonnen[4]. Doch erst im Juli 2008 kam es laut einer Pressemitteilung von Adobe zu einem nennenswerten Fortschritt dieser Zusammenarbeit[5][6].

Um die Suchmaschinen-Algorithmen anpassen zu können, war es notwendig die Lizenz-Bestimmungen der Flash-Technologie zu lockern. Bis zu diesem Zeitpunkt war es, außer für Adobe selbst, niemandem erlaubt eine Abspiel-Software für Flash-Inhalte zu entwickeln. Doch Adobe genehmigte nicht nur die Anfertigung individueller Flash Player, sondern veröffentlichte zusätzlich noch die Spezifikationen des Flash 8 und Flash 9 SWF-Dateiformats, um die Entwickler in ihrer Arbeit zu unterstützen.

4 vgl. Flash Player Developer Center, http://www.adobe.com/devnet/flashplayer/articles/swf_searchability.html?devcon=f1, aufgerufen am 2. Februar 2009

5 vgl. Adobe verbessert Suche in Rich Media-Inhalten, http://www.adobe.com/de/aboutadobe/pressroom/pr/jul2008/044.pdf, aufgerufen am 5. Februar 2009

6 vgl. Web Suchmaschinen Blog, http://www.at-web.de/blog/20080705/google-liest-flash-nun-besser-yahoo-will-folgen.htm, aufgerufen am 5. Februar 2009

4.2 Flex Compiler

Der Quellcode einer Flex-Anwendung kann nur mit dem Flex-Compiler[7] übersetzt werden. Das Flex-SDK beinhaltet neben dem Flex-Application-Compiler *mxmlc* noch andere Tools, die bei der Kompilierung der Anwendung behilflich sein sollen. In den folgenden Unterabschnitten wird genauer auf die einzelnen Kompilierwerkzeuge und deren Handhabung eingegangen.

4.2.1 Flex Application Compiler

Das eigentliche Herzstück des Flex-SDK ist der kostenlose Compiler *mxmlc*[8], der die ActionScript- und MXML-Dateien in eine SWF-Datei umwandelt. Das Kommandozeilenprogramm wird auch Application-Compiler genannt und bietet zahlreiche Optionen, um beispielsweise Bibliotheken einzubinden oder auch Debug-Informationen in die SWF-Datei einzubetten. Außerdem ist es auch möglich Eigenschaften der Applikation, wie die Framerate oder die Breite und Höhe der Anwendung, festzulegen. In Tabelle 4.1 werden einige der wichtigsten *mxmlc*-Optionen[9] und der deren Funktion aufgelistet. Der folgende Kommandozeilen-Befehl führt den mxmlc-Compiler aus.

```
mxmlc [options] target_file
```

Im Platzhalter `[options]` werden alle Compiler-Optionen gesetzt und `target_file` steht für den Dateinamen der zu erstellenden SWF.

7 vgl. Flex Compiler Shell, http://labs.adobe.com/wiki/index.php/Flex_Compiler_Shell, aufgerufen am 12. Februar 2009

8 vgl. Using mxmlc, the application compiler, http://livedocs.adobe.com/flex/3/html/help.html?content=compilers_13.html, aufgerufen am 12. Februar 2009

9 vgl. About the application compiler options, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/compilers_14.html#157203, aufgerufen am 12. Februar 2009

Option	Beschreibung
`debug=true|false`	Der Standardwert der Debug-Option ist false. Wird debug=true gesetzt, werden Debug-Informationen in die SWF-Datei kompiliert. Diese Informationen werden vom Kommandozeilen-Debugger oder vom Flex Builder Debugger verwendet. Mit Hilfe der Informationen können die Debugger beispielsweise die Quellcode-Zeile ausgeben, die einen Laufzeitfehler verursacht haben. Eine SWF-Datei mit Debug-Informationen ist in der Regel etwas größer.
`default-frame-rate int`	Legt die Framerate der Applikation fest. Der Standardwert ist 24.
`default-size width height`	Legt die Breite und Höhe der Applikation fest.
`includes class [...]`	Alle hier angegebenen Klassen werden in die resultierende SWF-Datei verlinkt, egal ob diese Klassen zum Zeitpunkt der Kompilierung genutzt werden oder nicht.
`include-libraries library [...]`	Alle hier angegebenen SWC-Datei werden in die resultierende SWF-Datei verlinkt, egal ob die Klassen in der SWC-Bibliothek zum Zeitpunkt der Kompilierung genutzt werden oder nicht.
`incremental=true|false`	Mit dieser Option kann die inkrementelle Kompilierung aktiviert werden. Der Flex Builder nutzt diese Option automatisch, um eine schneller Kompilierung zu gewährleisten. Auch der Web-basierte Compiler von Flex nutzt diese Option.
`keep-generated-actionscript =true|false`	Diese Option ist standardmäßig false. Wird keep-generated-actionscript auf true gesetzt, so werden alle ActionScript-Klassen, die temporär vom Compiler erzeugt werden, nach dem Kompiliervorgang nicht wieder gelöscht. Diese Option bietet sich vor allem für eine genaue Überprüfung der Ergebnisse des Compilers, bei der Übersetzung von MXML-Dateien an, da diese generell in ActionScript-Klassen übersetzt werden.
`optimize=true|false`	Wird `optimize=true` gesetzt, so wird nach der Kompilierung der Optimizer von Flex aktiviert. Der Optimizer verringert die Dateigröße und erhöht die Performance der erzeugten SWF-Datei, indem es den Bytecode der SWF-Datei überarbeitet.
`output filename`	Die output-Option definiert den Pfad und den Namen der vom Compiler erzeugten Datei. Wird diese Option weggelassen, erstellt der Compiler eine SWF-Datei mit dem Namen der ActionScript-Klasse

Tabelle 4.1: Ausgewählte Optionen des *mxmlc*-Compilers

4.2.2 Flex Component Compiler

Mit dem Kommandozeilen-Programm *compc*[10] ist dem Flex-Framework ein weiterer Compiler hinzugefügt worden. Der *compc*-Compiler wird auch als Component-Compiler bezeichnet und kann zur Erzeugung von SWC-Dateien genutzt werden. Das Programm ist dabei in der Lage, Quellcode, Bilder, Stylesheets und andere Asset-Dateien in eine SWC-Datei zu compilieren.

Der *compc*-Compiler verfügt über zahlreiche Parameter und unterstützt auch viele Optionen des *mxmlc*-Compilers. Metadaten-Optionen, wie Title oder Datum, sowie applikationsspezifische Informationen, wie `default-background-color` und `default-frame-rate`, werden allerdings nicht vom compc-Compiler unterstützt. Des Weiteren werden vom Komponenten-Compiler Einstellungen unterstützt, die nicht vom Applikations-Compiler verstanden werden. Einige dieser Optionen werden in Tabelle 4.2 aufgeführt.

Option	Beschreibung
`directory=false|true`	Mit der Option `directiory=true` wird der Compiler angewiesen die Inhalte der SWC-Datei im Open-Directory-Format auszugeben. Standardwert ist `false`, mit dem eine SWC-Datei geschrieben wird.
`include-classes class [...]`	Gibt die Klassen an, die in die SWC-Datei einkompiliert werden sollen.
`include-file name path [...]`	Fügt Dateien der SWC-Datei hinzu. Diese Option ist sehr hilfreich, um Grafikdateien hinzuzufügen, die in Stylesheets oder im MXML-Code genutzt werden.

Tabelle 4.2: Ausgewählte Optionen des *compc*-Compilers

4.2.3 Flex Compiler Shell

Eine Alternative zu den beiden bereits vorgestellten Programmen *mxmlc* und *compc* liegt dem Flex SDK noch das *fcsh*-Werkzeug[11] (Flex-Compiler-Shell) bei. Die Compiler-Shell ist in der Lage Applikationen, Module und Komponenten-Bibliotheken zu erstellen. Das Programm funktioniert ähnlich wie der mxmlc- und compc-Compiler, arbeitet jedoch bedeutend schneller als die beiden anderen Tools. Ein Grund für die schnellere Kompilieren ist das verbesserte Speichermanagement. So versucht die Shell häufig genutzte Daten im Hauptspeicher zu belassen, die bei jedem Kompiliervorgang von der Festplatte neu geladen werden müssten. Auch bereits erzielte Kompilierergebnisse versucht die Shell im Hauptspeicher

10 vgl. Using compc, the component compiler; Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/compilers_22.html#250507, aufgerufen am 14. Februar 2009

11 vgl. Using fcsh, the Flex compiler shell; Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/compilers_32.html#190522, aufgerufen am 19. Februar 2009

zu belassen, um weitere Kompilierungen zu verkürzen. Diese Technik wird als inkrementelle Kompilierung bezeichnet und bietet vor allem bei Projekten mit viel Quellcode einen enormen Geschwindigkeitszuwachs. Adobe empfiehlt daher die Verwendung des *fcsh*-Tools für sehr komplexe Anwendungen und für Projekte, bei denen besonders häufig kompiliert werden muss. Nach dem Aufruf von *fcsh* kann die Shell mit verschiedenen Kommandos gesteuert werden. Alle verfügbaren *fcsh*-Befehle werden in Tabelle 4.3 aufgelistet.

Kommando	Beschreibung
`clear [id]`	Löscht alle mittels ID(s) angegebenen Compiler-Targets. Werden keine IDs übergeben, werden alle Targets gelöscht.
`compile id`	Kompiliert ein Compiler-Target mit der gegebenen ID. Wird hier versucht ein Target zu kompilieren, bei dem keine Änderungen zur vorherigen Kompilierung festzustellen sind, ignoriert *fcsh* diesen Befehl.
`compc arg1 [...]`	Erzeugt eine SWC-Datei aus den angegebenen Sourcen. Dieser Befehl generiert außerdem ein Compiler-Target und gibt dessen ID zurück.
`info [id]`	Zeigt Informationen über die gegebene Compiler-Target-ID an.
`mxmlc arg1 [...]`	Kompiliert und optimiert eine Flex-Applikation oder -Modul aus den angegebenen Sourcen. Bei diesem Vorgang wird der mxmlc-Compiler verwendet, ein Compiler-Target generiert und dessen ID zurück gegeben.
`quit`	Beendet das *fcsh*-Tool. Alle temporären Dateien und genutzte Daten im Hauptspeicher werden gelöscht. Die in dieser Session verwendeten Compiler-Targets sind danach nicht mehr ansprechbar.

Tabelle 4.3: Ausgewählte Kommandos des *fcsh*-Tools

4.3 ActionScript 3.0

ActionScript 3.0 (kurz: AS3) ist eine objektorientierte Programmiersprache. ActionScript wurde ursprünglich von Macromedia entwickelt und Macromedia Flash wurde das erste Programm, das diese Programmiersprache unterstützte. Flash diente bis dahin nur zur Erstellung von Medieninhalten, wie Animationen und Filmen. Die Integration von ActionScript gab den Flashentwicklern die Möglichkeit interaktive Anwendungen zu erstellen. ActionScript zählt wie JavaScript zu den Scriptsprachen und basiert auf dem ECMA-Script-Standard 262. Macromedia, wie auch Adobe, haben sich aktiv an der Entwicklung des EcmaScript-Standards

beteiligt. Somit ist ActionScript bereits in der aktuellen Version 3 vollständig kompatibel zum nächsten Release des Standards.

Gegenüber seinen Vorgängerversion besticht ActionScript 3.0 besonders durch die strenge Typisierung. Hauptsächlich für den Flash-/Flex-Entwickler ergeben sich dadurch enorme Verbesserungen. Zum einen können nun Quelltexte besser strukturiert und organisiert werden, welche die Lesbarkeit des Codes unterstützen. Des Weiteren werden dem Entwickler bei AS3 hilfreichere Fehlermeldungen angezeigt, die ein schnelleres Debugging der Anwendung ermöglichen. Hinsichtlich der Syntax näherte sich AS3 stark C# und Java, das den Umstieg zwischen den Programmiersprachen sehr erleichtert.

Aufgrund der tief greifenden Änderungen am SWF-Dateiformat und dessen „Abspielens“ war die Entwicklung einer neuen Laufzeitumgebung nötig. Der Flash-Player 9 basierte aus diesem Grund nicht mehr auf der VM1 (Virtual Machine 1 oder auch AVM1 für ActionScript Virtual Machine) wie seine Vorgängerversionen, sondern auf der neu entwickelten VM2. Bei der Entwicklung der neuen Virtual Machine wurde insbesondere auf eine höhere Geschwindigkeit beim Abspielen von Flashfilmen und auf eine größere Skalierbarkeit der Anwendungen geachtet. Der neue integrierte Just-In-Time-Compiler (JIT Compiler) wandelt zur Laufzeit AS3 Bytecode in nativen Code um, das nochmals die Ausführgeschwindigkeit erhöht. Ergebnis der Optimierungen ist eine bis zu 10 mal schnellere Ausführung von ActionScript-Code bei einem geringerem Speicherverbrauch. Wie in den Vorgängerversionen wurde auch im Flash-Player 9 eine vollständige Abwärtskompatibilität eingebaut. Dies wird mit einer integrierten VM1 gewährleistet, die parallel zur VM2 läuft und für Flash Versionen 1 bis 8 genutzt wird.

Adobe veröffentlichte die ActionScript-VM2 vollständig als OpenSource, indem sie das Projekt der Mozilla Foundation übergaben, die auch den Mozilla Firefox als OpenSource entwickeln und verwalten. Mit diesem Schritt erhofft sich Adobe, dass die AVM2-Technologie als Standard übernommen wird, um zukünftig eine Kompatibilität mit JavaScript zu gewährleisten.

4.4 MXML

Mit Flex 1.0 wurde erstmals die Programmiersprache MXML vorgestellt. MXML, oder auch Flex Markup Language genannt, ist eine XML-basierte, deklarative Programmiersprache[12]. Ähnlich wie HTML wird MXML eingesetzt, um das Layout einer Benutzer-Oberfläche eines Programmes zu definieren. Gegenüber HTML zeichnet sich MXML durch seine besser strukturierten und klareren Syntax aus. Des Weiteren beinhaltet MXML eine weitaus größere

12 vgl. An overview of MXML, The Flex markup language; http://www.adobe.com/devnet/flex/articles/paradigm.html, aufgerufen am 20. Februar 2009

Anzahl Tags als HTML. Darüber hinaus hat der Programmierer die Möglichkeit auf Basis der vorhandenen Komponenten eigene Tags zu definieren.

Die Entwickler von Flex haben MXML nicht nur für die visuelle Gestaltung der Benutzeroberfläche einer Anwendung vorgesehen. Auch andere wichtige Aspekte eines Programms können mithilfe von MXML-Tags deklariert werden. So können beispielsweise Animationen für die Zustände der Benutzeroberfläche mittels MXML festgelegt werden. Aber auch für Datenanbindungen, wie zum Beispiel zu Webservices oder XML-Dateien, stehen Komponenten zur Verfügung. Selbst komplexe Datenstrukturen können mittels MXML-Tags definiert werden.

So wie andere Markup-Sprachen verfügt auch MXML nur über wenige Möglichkeiten die Programmlogik zu beschreiben. Das Flex-Framework stellt für die Erstellung eine interaktiven Anwendung daher eine Kombination aus MXML und ActionScript zur Verfügung. Die problemlose Einbettung von ActionScript-Code, selbst innerhalb einer MXML-Datei, erlauben es auf Interface-Ereignisse zu reagieren. In Kapitel 4.4.2 wird etwas genauer auf die Integrierung von ActionScript in MXML-Code eingegangen.

Im Unterabschnitt 4.4.1 werden einige MXML-Komponenten vorgestellt, um einen kleinen Einblick in die grafischen Möglichkeiten von Flex 3 zu erhalten. In den folgenden Abschnitten werden weitere Besonderheiten der Programmiersprache vorgestellt.

4.4.1 Visuelle Komponenten

Das Flex-Framework stellt dem Programmierer eine immense Anzahl von Interface-Komponenten zur Verfügung. Natürlich beinhaltet das Framework dabei auch die Standard-Komponenten, wie Texteingabefelder, Checkboxen, Comboboxen oder einfache Buttons. Doch das bemerkenswerte an Flex sind die erweiterten und meist gut durchdachten komplexeren Komponenten.

Containerkomponenten

Die auf den ersten Blick weniger komplexen, aber dennoch recht mächtigen Komponenten, stellen die Container-Elemente von Flex 3 dar. Container helfen dem Programmierer auf einfachsten Wege Layout-Richtlinien festzulegen. So ist es möglich mehrere Komponenten innerhalb eines Containers relativ zueinander zu Positionieren, d. h keine genauen Position festzulegen. Beispielsweise werden alle Komponenten, die sich in einem `HBox`-Tag befinden, horizontal ausgerichtet. Aber auch eine absolute Positionierung von grafischen Elementen, wie es ein Flash-Entwickler gewohnt ist, ist durchaus mittels eines `Canvas`-Objektes möglich. Alle Elemente innerhalb eines `Canvas` können mit x- und y-Koordinaten positioniert werden, wodurch auch eine Überschneidung der Elemente möglich wird. In Listing 4.1 zeigt ein

Code-Beispiel, wie Flex-Komponenten innerhalb eines Canvas-Objektes positioniert werden können. Ein Screenshot der erstellten Flex-Anwendung ist in Abbildung 4.1 zu sehen.

```
1 ...
2         <mx:Canvas>
3                 <mx:Image x="30" y="30" source="assets\fx_icon.gif" />
4                 <mx:Image x="20" y="50" source="assets\dw_icon.gif" />
5                 <mx:Image x="60" y="90" source="assets\fl_icon.gif" />
6         </mx:Canvas>
7 ...
```

Listing 4.1: Positionierung mehrerer Komponenten innerhalb eines Canvas-Objektes mit Hilfe von x- und y-Koordinaten

Abbildung 4.1: Screenshot der kompilierten Anwendung mit Quellcode aus Listing 4.1

Buttonkomponenten

Flex stellt dem Entwickler vier verschiedene Buttontypen zur Verfügung:

- den normalen `Button`,
- den `LinkButton`, wie man ihn von HTML-Internetseiten kennt
- den `RadioButton`, der eine Single-Choice Optionswahl ermöglicht
- und die `CheckBox`, die eine Multiple-Choice Optionswahl darstellt

Abbildung 4.2: Buttonskomponenten `Button`, `Label`, `RadioButton` und `CheckBox`

Textkomponenten

Flex unterstützt fünf Typen von Text-Komponenten. Weitergehend können diese Typen in editierbare und nicht editierbare Text-Komponenten kategorisiert werden. Zu den nicht-editierbaren Texten gehören das `Label` und die `Text`-Komponente. Die `Label`-Komponente bietet sich für einzeilige Texte an, wobei die `Text`-Komponente für mehrzeiligen Text geeignet ist.

Die Komponenten `TextInput` und `TextArea` sind editierbare Text-Komponenten, d. h. der Nutzer hat hier die Möglichkeit eigenen Text per Keyboard einzugeben. `TextInput` stellt dabei das editierbare Gegenstück eines `Label` sowie `TextArea` das Pendant zur `Text`-Komponente dar.

Diese vier genannten Komponenten gehören zum Standard für interaktive Benutzeroberflächen und sind auch in vielen anderen Skriptsprachen, wie zum Beispiel HTML vorhanden. Ein weitaus komplexeres Flex-Anzeigeobjekt stellt die `RichTextEditor`-Komponente dar. Sie ermöglicht den Nutzern die Eingabe und Formatierung von Text. Die Texteigenschaften, die der Nutzer dabei ändern kann, umfassen unter anderem Schriftart, -größe, -farbe und -stil sowie Textausrichtung, Aufzählungszeichen und die Festlegung von Hyperlinks. Abbildung 4.3 zeigt eine Instanz der RichTextEditor-Komponentenklasse.

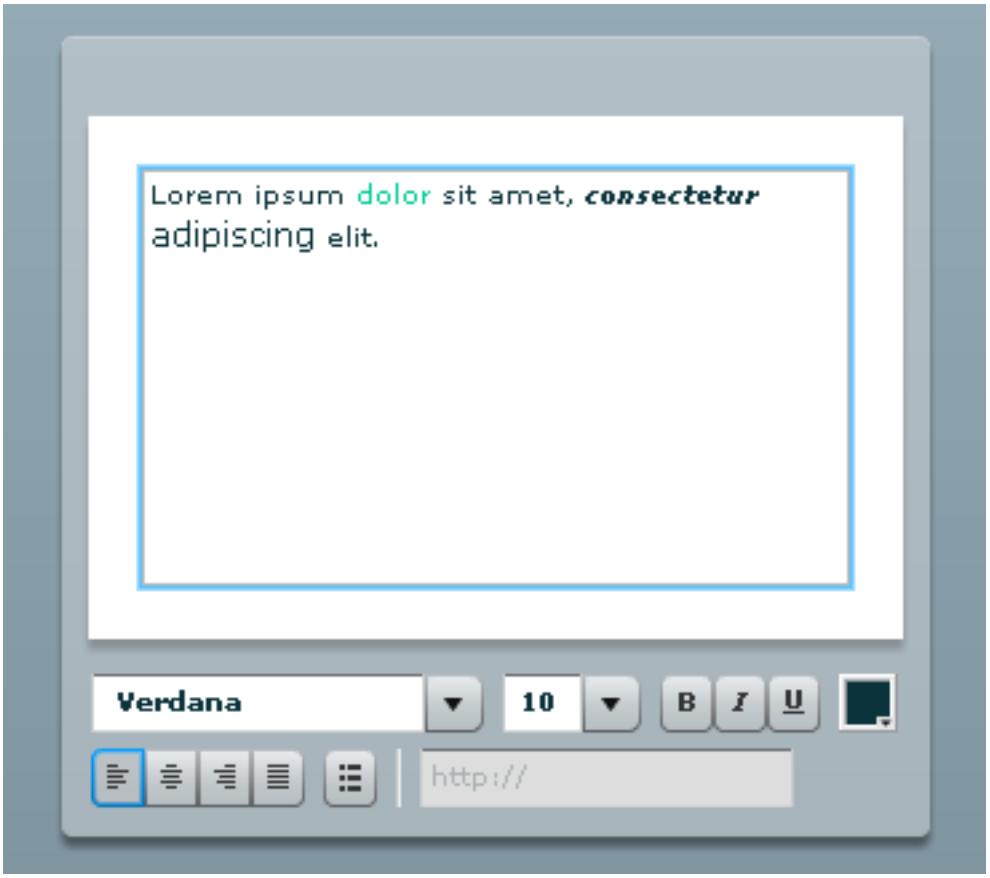


Abbildung 4.3: `RichTextEditor`-Instanz

Listenkomponenten

Die anspruchsvollsten Komponenten von Flex sind die Listen-basierenden Kontrollelemente. Diese Komponenten ermöglichen dem Nutzer, ein oder mehrere Optionen aus einer Liste von Optionen auszuwählen. In ihrer einfachsten Variante besteht eine Liste aus einer vertikalen, scrollbaren Liste von Textfeldern. Davon ausgehend kann eine Liste bedeutend erweitert werden. So kann sie beispielsweise auch aus mehreren Spalten bestehen, einen rasterbasierten Aufbau haben, zusammenklappbare Inhalte besitzen, oder wiederum Listen beinhalten. Bild 4.4 zeigt die erweiterten Listen-Kontrollelemente `DataGrid`, `HorizontalList`, und `TileList`.

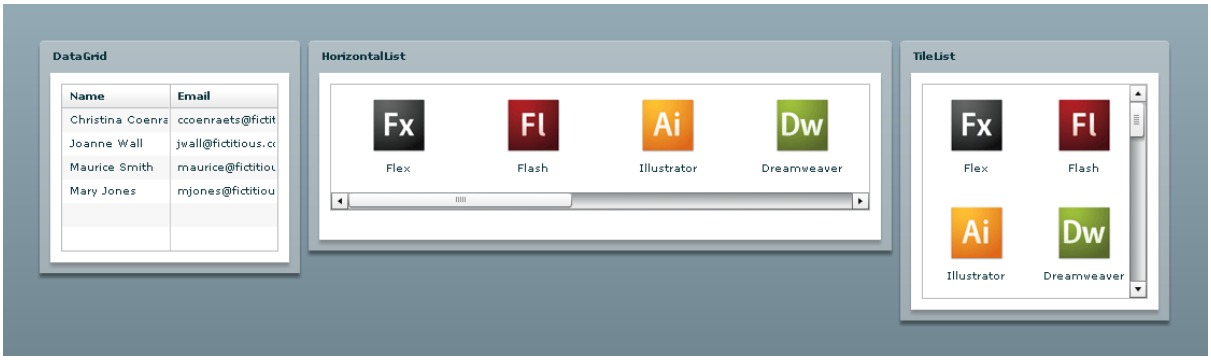


Abbildung 4.4: v.l.n.r.: `DataGrid` (Listendarstellung von Texten), `HorizontalList` (horizontale Anordnung von Komponenten), `TileList` (Anordnung von Objekten gleicher Größe in einem Raster)

Um eine Liste mit Daten zu füllen, wird ein Data-Provider benötigt. Ein Data-Provider stellt das Datenmodell der Liste dar. Jede Listenkomponente besitzt eine *dataProvider*-Eigenschaft, um eine Datenquelle für die Liste festlegen zu können. Typische Datenquellen für eine Liste sind XML-Daten oder Daten im ArrayCollection-Format.

Navigationskomponenten

Eine der wichtigsten Elemente einer Anwendung sind die Möglichkeiten, mit die der Benutzer zwischen verschiedenen Ansichten, Seiten oder Optionen navigieren kann. Flex bietet mehrere Möglichkeiten um eine Navigation innerhalb einer Flex-Applikation zu gewährleisten. Typische und oft verwendete Navigations-Kontrollelemente sind Akkordeonmenüs, Menüleisten oder Tabmenüs.

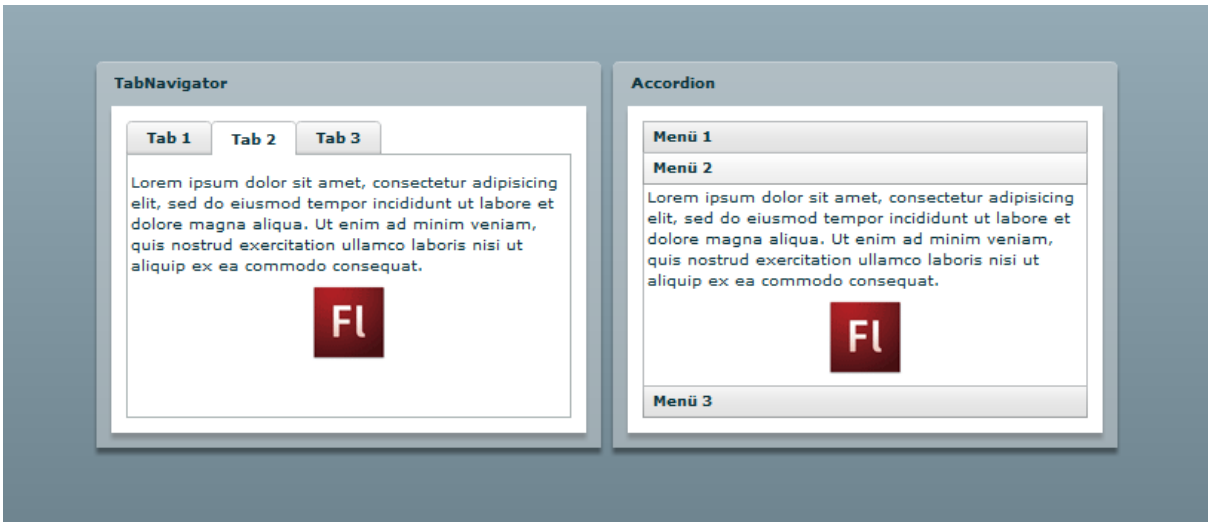


Abbildung 4.5: Navigations-Kontrollelemente `TabNavigator` und `Accordion`

4.4.2 Integration von ActionScript

Die Flex-Architektur sieht vor, möglichst alle visuellen Komponenten einer Anwendung durch MXML umzusetzen und jegliche Programmlogik mit ActionScript zu realisieren. Flex bietet dem Programmierer mehrere Wege an, ActionScript in einer Applikation zu nutzen. In den folgenden Abschnitten sind vier Möglichkeiten zur Integration von ActionScript in eine Flex-Applikation aufgelistet — begonnen bei der simpelsten bis hin zur komplexesten Variante.

Inline-ActionScript

Inline-ActionScript bietet sich hervorragend für kurze ActionScript-Befehle, Data-Binding oder Event-Handling an. Das Codebeispiel in Listing 4.2 zeigt ein sehr einfaches Beispiel von Inline-ActionScript.

```
1 ...
2 <mx:Button id="button" label="Show Trace" click="trace('Hallo Welt')" />
3 ...
```

Listing 4.2: Flex-Button mit Click-Event und Inline-ActionScript

Im Beispiel wird eine Button-Instanz mit dem Label „Show Trace“ und der ID „button“ erzeugt. Dem Button-Attribut „click“ kann ein EventHandler übergeben werden, der bei jedem Mausklick auf den Button ausgeführt wird. Flex erlaubt an dieser Stelle nicht nur einen EventHandler anzuhängen sondern gleich hier ActionScript-Code zu platzieren. So wird in diesem Beispiel bei jedem Klick „Hallo Welt“ per trace-Funktion ausgegeben.

Die Verwendung von Inline-ActionScript wie in Beispiel 4.2 besitzt den Nachteil, dass der Code bei der Kompilierung statisch eingefügt wird und beispielsweise keine Ausgabe von Variablen ermöglicht. Diese Beschränkung kann man mit dem sogenannten *Data-Binding* umgehen. *Data Binding* kommt zum Einsatz, sobald geschweifte Klammern innerhalb von MXML-Code genutzt werden. Im Gegensatz zu Inline-ActionScript ohne DataBinding wird der ActionScript-Code innerhalb der geschweiften Klammern erst zur Laufzeit des Programms ausgewertet.

Im Listing 4.3 wird ein sehr einfaches Beispiel von DataBinding gezeigt. In Zeile 3 wird ein Eingabefeld mit der ID „input“ erzeugt. Weiterhin wird in Zeile 4 ein Ausgabefeld mit der ID „output“ erstellt. Dem Ausgabefeld wird per DataBinding der Text des Eingabefeldes übergeben. Das DataBinding bewirkt nun, dass der eingegebene Text im Feld „input“ sofort im Textfeld „output“ ausgegeben wird. Detailliertere Betrachtungen der DataBinding-Technik werden im Unterabschnitt 4.4.3 vorgenommen.

```
...
<mx:VBox>
    <mx:TextInput id="input" />
    <mx:Text id="output" text="{input.text}" />
</mx:VBox>
...
```

Listing 4.3: DataBinding zwischen einer TextInput- und einer Text-Komponente

Eingebettetes ActionScript

Eine Einbindung von ActionScript kann auch mittels MXML-Tags durchgeführt werden. Mit dieser Technik lassen sich Eigenschaften wie auch Event-Handler an Komponenten übergeben. Der eingefügte ActionScript-Code muss dabei in CDATA-Blöcke gekapselt werden. Ein simples Beispiel für eingebettetes ActionScript zeigt Listing 4.4.

```
...
<mx:Button>
    <mx:click>
        <![CDATA[
            trace('Hallo Welt');
        ]]>
    </mx:click>
</mx:Button>
...
```

Listing 4.4: Flex-Button mit Click-Event

MXML-Script

Das Script-Element von MXML bietet eine weitere Möglichkeit ActionScript in MXML-Code zu integrieren. In Listing 4.5 ist ein Beispiel eines Script-Blocks zu sehen, in dem die Funktion *zeigeBeispiel()* definiert wird. Da der ActionScript-Code Sonderzeichen enthalten kann, die vom Flex-Compiler interpretierte werden und somit zu Fehlern führen könnten, muss der Quelltext wie beim eingebetteten ActionScript in einen CDATA-Block gekapselt werden.

```
...
<mx:Script>
<![CDATA[
    private function zeigeBeispiel( ):void {
        trace("Hallo Welt");
    }
]]>
</mx:Script>
...
```

Listing 4.5: MXML-Script mit Funktionsdeklaration

Die Script-Komponente ermöglicht noch eine Alternative zum Einbetten von ActionScript innerhalb eines Script-Tags. Listing 4.6 zeigt, wie man mit Hilfe des source-Attributs auch

ganze ActionScript-Dateien laden und integrieren kann. Mit Hilfe von MXML-Script können Klassen importiert und Eigenschaften bzw. Methoden deklariert werden.

```
1 ...
2 <mx:Script>
3 <![CDATA[
4     private function zeigeBeispiel( ):void {
5         trace("Hallo Welt");
6     }
7 ]]>
8 </mx:Script>
9 ...
```

Listing 4.6: MXML-Script mit source-Attribut

ActionScript-Klassen

ActionScript 3 ist eine vollständig objektorientierte Programmiersprache und unterstützt die Erstellung von Klassen. Neben der Einbettung von ActionScript mit den bereits genannten Möglichkeiten, ist die Verwendung von Klassen die mächtigste Variante und wird allgemein empfohlen[13]. Auch Adobe empfiehlt die Verwendung von eigenen ActionScript-Klassen, um einen möglichst gut strukturierten Programmaufbau zu erhalten[14].

4.4.3 Data-Binding

Data-Binding beschreibt den Prozess, die Daten eines Objektes mit den Daten eines weiteren Objektes zu verknüpfen. Diese Technik wird von den verschiedensten Programmiersprachen und Frameworks unterstützt[15]. Grundsätzlich dient Data-Binding dazu, eine Verbindung zwischen Benutzeroberfläche und Businesslogik einer Applikation herzustellen. Die Richtung in die die Daten dieser Verbindungen fließen, hängt vom Typ des Data-Bindings ab. Jede Programmiersprache, die Data-Binding unterstützt, unterstützt unter Umständen nicht alle Varianten dieser Technik oder stellt gar eigene Varianten von Data-Binding dem Programmierer zur Verfügung.

Die Verknüpfung der Daten in eine Richtung (One-Way-Data-Binding) bindet die Daten einer Quelle an ein Ziel-Objekt. Ändern sich die Daten der Quelle, so wird automatisch das Ziel-Objekt aktualisiert. Werden allerdings Daten des Ziel-Objektes beispielsweise durch Texteingaben geändert, so wird das vom Quellobjekt nicht registriert. Eine solche bidirektionale

13 vgl. O'Reilly Programming Flex 2 (engl. Ausgabe), Seite 57

14 vgl. Einfache ActionScript 3.0-Klasse erstellen, Adobe Flash-Kurzanleitungen, http://www.adobe.com/de/devnet/flash/quickstart/creating_class_as3/, aufgerufen am 20. Februar 2009

15 vgl. Data Binding Overview, Microsoft .NET Developer Center, http://msdn.microsoft.com/en-us/library/ms752347.aspx, aufgerufen am 21. Februar 2009

Verknüpfung der Daten stellt das Two-Way-Data-Binding bereit. Ein einfacher Anwendungsfall von Two-Way-Data-Binding[16] ist die Verbindung zweier Textfelder. Werden Daten in Textfeld A eingegeben, so werden sie auch in Textfeld B angezeigt. Bearbeitet der Nutzer Textfeld B, so wird der Text wiederum in Textfeld A dargestellt.

In Flex 3 wird Data-Binding als eine der herausragendsten Features genannt und sollte laut Adobe intensiv genutzt werden. Flex 3 unterstützt aber lediglich One-Way-Data-Binding[17], wobei mit etwas Programmieraufwand durchaus Two-Way-Data-Binding genutzt werden kann. Data-Binding kann in Flex 3 über drei verschiedene Wege genutzt werden. Die folgenden Beispiele, die von der Adobe Website entnommen wurden, zeigen die Möglichkeiten, wie Data-Binding in Flex genutzt werden kann.

Data-Binding mittels geschweifter Klammern

Durch die Verwendung geschweifter Klammern innerhalb von MXML-Ausdrücken wird automatisch Data-Binding aktiviert. Listing 4.7 zeigt ein Eingabefeld mit der ID *myTI* und eine Text-Komponente mit der ID *myText*. Das Data-Binding wird in Zeile 5 definiert, indem der text-Eigenschaft der Text-Komponente der Text des TextInput-Felds übergeben werden. Sobald sich nun die text-Eigenschaft von *myTI* ändert, kopiert Flex automatisch den aktuellen Wert *myTI.text* zum Ziel *myText.text*.

```
1 <?xml version="1.0"?>
2 <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
3
4     <mx:TextInput id="myTI" text="Enter text here"/>
5     <mx:Text id="myText" text="{myTI.text}"/>
6
7 </mx:Application>
```

Listing 4.7: Data-Binding in MXML mittels geschweifter Klammern

Flex hat die ursprüngliche Definition von Data-Binding, die nur ein Kopieren der Werte von einer Quelle zu einem Ziel umfasst, erweitert. Dadurch sind auch ActionScript-Ausdrücke innerhalb der geschweiften Klammern möglich, die bei jeder Aktualisierung der Quelle ausgeführt werden. Listing 4.8 veranschaulicht die Möglichkeiten, die dem Entwickler durch diese Technik gegeben werden, an Hand eines kleinen Beispiels. Das Beispiel zeigt fast den gleichen Code wie in Listing 4.7, nur dass hier zusätzlich die ActionScript-Methode `String.toUpperCase()` zum Einsatz kommt. So wird jegliche Eingabe in Textfeld *myTI* in Großbuchstaben konvertiert und in myText angezeigt. Die Nutzung von ActionScript innherhalb von geschweiften Klammern unterliegen aber gewissen Beschränkungen, auf

16 vgl. Two-way Data Binding; Adobe Open Source: http://opensource.adobe.com/wiki/display/flexsdk/Two-way+Data+Binding, aufgerufen am 2. März 2009

17 vgl. Binding Data, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/help.html?content=databinding_1.html, aufgerufen am 2. März 2009

die in dieser Arbeit nicht weiter eingegangen wird. Weitere Informationen über diese Einschränkungen können in der Flex 3 Hilfe nachgelesen werden[18][19].

```
1 <?xml version="1.0"?>
2 <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
3
4     <mx:TextInput id="myTI"/>
5     <mx:Text id="myText" text="{myTI.text.toUpperCase()}"/>
6
7 </mx:Application>
```

Listing 4.8: Data-Binding in MXML mittels geschweifter Klammern

Data-Binding mittels Binding-Tag

Als Alternative zu geschweiften Klammern, kann das <mx:Binding>-Tag eingesetzt werden, um Data-Binding in MXML-Code zu nutzen. Listing 4.9 zeigt die Syntax des <mx:Binding>-Tag. Der Eigenschaft source wird *myTI* als Quelle und destination *myText* als Ziel des Data-Bindings übergeben. Wie in Beispiel 4.7 wird auch hier jeglicher Text, der in *myTI* eingegeben wird in *myText* wiedergegeben. Im Gegensatz zum Data-Binding mittels geschweifter Klammern kann das <mx:Binding>-Tag zur vollständigen Separierung von Benutzeroberfläche und Datenmodell verwendet werden. Des Weiteren können mithilfe von mehreren <mx:Binding>-Tags mehrere Datenquellen an ein Zielobjekt gebunden werden[20]. Listing 4.10 zeigt ein Data-Binding von zwei Eingabefeldern *input1* und *input2* als Datenquelle zu dem TextArea *myTA*.

```
1 <?xml version="1.0"?>
2 <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
3
4     <mx:TextInput id="myTI"/>
5     <mx:Text id="myText"/>
6     <mx:Binding source="myTI.text" destination="myText.text"/>
7
8 </mx:Application>
```

Listing 4.9: Data-Binding in MXML mittels Binding-Komponente

18 vgl. Using an E4X expression in a data binding expression; Adobe Flex 3 Help: http://livedocs.adobe.com/flex/3/html/help.html?content=databinding_6.html#189734, aufgerufen am 2. März 2009

19 vgl. Using ActionScript in data binding expressions, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/help.html?content=databinding_5.html#189605, aufgerufen am 2. März 2009

20 vgl. Using data binding with data models; Adobe Flex 3 Help: http://livedocs.adobe.com/flex/3/html/help.html?content=databinding_3.html#189940, aufgerufen am 7. März 2009

```
1  <?xml version="1.0"?>
2  <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
3
4      <mx:Binding source="input2.text" destination="myTA.text"/>
5
6      <mx:TextInput id="input1"/>
7      <mx:TextInput id="input2"/>
8
9      <mx:TextArea id="myTA" text="{input1.text}"/>
10
11 </mx:Application>
```

Listing 4.10: Data-Binding von zwei Quellen an ein Ziel-Objekt mittels Binding-Komponente und geschweiften Klammern

Data-Binding mittels ActionScript

Geschweifte Klammern und das `<mx:Binding>`-Tag definieren beide ein Data-Binding zum Zeitpunkt der Kompilierung. Mit Hilfe von ActionScript ist auch die Definition von Data-Bindings zur Laufzeit möglich. In Listing 4.11 wird ein Data-Binding mithilfe der statischen Methode `BindingUtils.bindProperty()` definiert. Wie in den vorherigen Beispiel wird wieder ein Texteingabefeld mit einer Text-Komponente verknüpft. In diesem Beispiel musste das Data-Binding zum *preinitialize*-Event der Komponente definiert werden, da Flex alle Data-Bindings zum *initialize*-Event auslöst und somit der Startwert der Quellobjekte von den Zielobjekten übernommen werden. Die Flex-BindingUtils-Klasse bietet außerdem noch die statische Methode `BindingUtils.bindSetter()`, um eine ActionScript-Funktion an eine Datenquelle zu binden. Listing 4.12 zeigt die Verwendung von `BindingUtils.bindSetter()` in Zeile 18. Bei jeder Änderung des Textes in *myTI* wird die Funktion `updateMyString()` aufgerufen in der der eingegebene Text zu Großbuchstaben mittels der `String.toUpperCase()`-Methode gewandelt wird und der TextArea-Komponente *myTA* übergeben wird.

```
1  <?xml version="1.0"?>
2  <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
3
4      <mx:Script>
5        <![CDATA[
6          import mx.binding.utils.*;
7
8          // Define data binding.
9          public function initBindingHandler():void {
10             BindingUtils.bindProperty(myText, "text", myTI, "text");
11         }
12       ]]>
13     </mx:Script>
14
15     <mx:TextInput id="myTI"/>
16     <mx:Text id="myText" preinitialize="initBindingHandler();"/>
17
18 </mx:Application>
```

Listing 4.11: Data-Binding mittels ActionScript 3

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
        <![CDATA[

            import mx.binding.utils.*;
            import mx.events.FlexEvent;

            // Method called when myTI.text changes.
            public function updateMyString(val:String):void {
                myTA.text = val.toUpperCase();
            }

            <!-- Event listener to configure binding. -->
            public function mySetterBinding(event:FlexEvent):void {
                var watcherSetter:ChangeWatcher =
                    BindingUtils.bindSetter(updateMyString, myTI, "text");
            }
        ]]>
    </mx:Script>

    <mx:Label text="Bind Setter using setter method"/>
    <mx:TextInput id="myTI" text="Hello Setter" />
    <mx:TextArea id="myTA" initialize="mySetterBinding(event);"/>
</mx:Application>
```

Listing 4.12: Data-Binding mittels BindingUtils.bindSetter()

Bidirektionales Data-Binding

Wie zu Beginn von Unterabschnitt 4.4.3 beschrieben, unterstützt Flex kein natives bidirektionales Data-Binding. Listing 4.13 zeigt wie dennoch eine beidseitige automatische Aktualisierung der Werte erreicht werden kann. Dazu werden zwei „normale“ Data-Bindings verwendet, die jeweils das andere Eingabefeld aktualisieren.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:TextInput id="input1" text="{input2.text}"/>
    <mx:TextInput id="input2" text="{input1.text}"/>

</mx:Application>
```

Listing 4.13: Bidirektionales Data-Binding mittels BindingUtils.bindSetter()

4.4.4 Datenstrukturen

Flex unterstützt die Erstellung von sogenannten „Flex-Data-Models“. Ein Flex-Data-Model ist ein MXML-Object, das zum Speichern und Manipulieren von Datensätzen genutzt werden kann. Ein einfaches Beispiel für ein Flex-Data-Model ist in Listing 4.14 zu sehen, das von der Adobe Flex Builder Hilfe[21] entnommen wurde. In diesem Beispiel werden Informationen

21 vgl. Flex data models, Flex Builder Hilfe, http://livedocs.adobe.com/flex/1/flex_builder_en/wwhelp/wwhimpl/common/html/wwhelp.htm?context=Using_Flex_Builder

über eine Person, wie dessen Namen, Alter und E-Mail-Adresse, strukturiert dargestellt und gespeichert.

```
<mx:Model id="person">
    <name>
        <first>John</first>
        <last>Doe</last>
    </name>
    <department>Accounting</department>
    <email>jdoe@goodcompany.com</email>
</mx:Model>
```

Listing 4.14: Flex-Data-Model

Ein Flex-Data-Model erlaubt aber nicht nur statische Daten, sondern unterstützt auch eine dynamische Daten-Anbindung zu anderen Objekten mittels Data-Binding. Des Weiteren können auch XML-Dateien als Daten-Modell eingebunden werden (Listing 4.15).

```
...
<mx:Model source="http://www.somesite.com/companyinfo.xml"
    id="myCompany"/>
...
```

Listing 4.15: Beispiel für dynamisches Laden eines Flex-Data-Models

4.4.5 Styles

Das visuelle Erscheinungsbild von Flex-Komponenten kann fast vollständig mit Hilfe von Style-Eigenschaften angepasst werden. Diese Eigenschaften definieren beispielsweise die Schriftart und -größe, die in einer Textkomponente verwendet wird. Einige Flex-Komponenten besitzen Style-Eigenschaften, die auf deren Kind-Komponenten vererbt werden[22]. Außerdem können nicht nur Styles einer Komponente, sondern auch ganzer Komponenten-Klassen definiert werden. Eine Änderung der Hintergrundfarbe der Button-Komponenten-Klasse würde somit bewirken, dass jeder Button innerhalb der Applikation diese Hintergrundfarbe übernimmt. Die globalen Styles[23] einer Komponente können zusätzlich dazu noch individuelle Styles erhalten. Dies soll laut Adobe[24] der Individualisierung der Benutzeroberfläche eine erhöhte Flexibilität bieten.

&file=brady713.htm#118986, aufgerufen am 7. März 2009

22 vgl. Applying styles and skins, Adobe Livedocs, http://livedocs.adobe.com/flex/3/html/help.html?content=working_comps_5.html - Styles mit Flex Builder, aufgerufen am 12. März 2009

23 vgl. Setting global styles, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/styles_02.html#161684, aufgerufen am 7. März 2009

24 vgl. Using Cascading Style Sheets, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/help.html?content=styles_03.html#190648, aufgerufen am 7. März 2009

Flex 3 bietet mehrere Möglichkeiten die Style-Eigenschaften von Komponenten festzulegen. So bieten manche Vorgehensweisen eine sehr spezifische Festlegung von Styles und können programmatisch zur Laufzeit zugewiesen werden. Andere Varianten sind weniger flexibel, aber benötigen weniger Systemressourcen.

Im Folgenden werden diese verschiedenen Wege genauer betrachtet. Außerdem wird kurz auf Cascading Stylesheets (CSS) in Flex eingegangen.

Externe Stylesheets

Die Flex-Komponente `<mx:Style>` ermöglicht ein Einbinden einer externen Stylesheet-Datei[25]. Wird eine externe Stylesheet-Datei eingebunden, so werden die darin befindlichen Style-Definitionen für die komplette Applikation übernommen und angewendet. Zu beachten ist, dass das eingebundene Stylesheet zur vom Compiler in die SWF- oder SWC-Datei einkompiliert wird und nicht zur Laufzeit geladen wird. Beispiel 4.16 zeigt den Quellcode zum Einbinden einer Styledatei mit dem Namen *myStyle.css*.

```
1 ...
2 <mx:Style source="myStyle.css"/>
3 ...
```

Listing 4.16: Einbindung einer externen Stylesheet-Datei

Lokale Style-Definitionen

Lokale Style-Definitionen[26] umfassen all die Styles, die nur innerhalb des Dokumentes gültig sind, in denen sie definiert wurden. Auch für diese Aufgabe ist die Flex-Komponente `<mx:Style>` vorgesehen. Listing 4.17 zeigt dazu ein Syntax-Beispiel für die Definition einer CSS-Klasse mit dem Namen `.myFontStyle`. Die Klasse definiert eine Schriftgröße von 15 Pixel und eine Schriftfarbe `#FF0000` (rot). In Zeile 8 wird die Style-Klasse dem Button mit der ID *myButton* übergeben. In der darauf folgenden Zeile wird eine Button definiert, dem keine gesonderte Styleklasse übergeben wird. Ein Screenshot der kompilierten Anwendung ist in Bild 4.6 zu sehen.

25 vgl. Using external style sheets; Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/help.html?content=styles_05.html#165272, aufgerufen am 7. März 2009

26 vgl. Using Cascading Style Sheets; Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/help.html?content=styles_05.html#165272, aufgerufen am 7. März 2009

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
    <mx:Style>
        .myFontStyle {
            fontSize: 15; color: #FF0000;
        }
    </mx:Style>
    <mx:Button id="myButton" styleName="myFontStyle" label="Click Me"/>
    <mx:Button id="myButton2" label="Click Me"/>
</mx:Application>
```

Listing 4.17: Definition einer lokalen Style-Klasse

Abbildung 4.6: Screenshot der Anwendung aus Listing 4.17

Listing 4.18 zeigt ein Beispiel, wie ein Style für alle Instanzen einer Flex-Komponente innerhalb eines Dokumentes definiert werden kann. Die Style-Klasse muss dafür den Namen der Komponenten-Klasse haben. In diesem Beispiel wird die Schriftgröße und -farbe der Button-Klasse definiert. Ein Screenshot der erzeugten SWF ist in Bild 4.7 zu sehen.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
    <mx:Style>
        Button {
            fontSize: 15;
            color: #FF0000;
        }
    </mx:Style>
    <mx:Button id="myButton" label="Click Me"/>
    <mx:Button id="myButton2" label="Click Me"/>
</mx:Application>
```

Listing 4.18: Definition einer lokalen Style-Klasse für Button-Komponenten

Abbildung 4.7: Screenshot der Anwendung aus Listing 4.18

StyleManager-Klasse

Die StyleManager-Klasse[27] ermöglicht eine weitere Variante Style-Definitionen für alle Instanzen einer Flex-Komponente festzulegen. In Listing 4.19 wird das Quellcode-Beispiel 4.17 mithilfe der StyleManager-Klasse umgesetzt. Das visuelle Ergebnis ist bei beiden Versionen das gleiche und kann in Bild 4.6 betrachtet werden. Die StyleManager-Klasse umfasst noch zahlreiche weitere Methoden, die an dieser Stelle nicht vorgestellt werden. Unter anderem ist es mit dieser Klasse möglich Style-Definitionen zur Laufzeit zu definieren und zuzuweisen.

```
1  <?xml version="1.0"?>
2  <mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
3      creationComplete="initApp()">
4      <mx:Script>
5          public function initApp():void {
6              StyleManager.getStyleDeclaration("Button")
7                  .setStyle("fontSize",15);
8              StyleManager.getStyleDeclaration("Button")
9                  .setStyle("color",0xFF0000);
10         }
11     </mx:Script>
12     <mx:Button id="myButton" label="Click Me"/>
13     <mx:Button id="myButton2" label="Click Me"/>
14 </mx:Application>
```

Listing 4.19: Style-Definition mit Hilfe der StyleManager-Klasse

Methoden getStyle() und setStyle()

Eine der flexibelsten Varianten, die Styles von Flex-Komponenten zu ändern, bieten die Methoden `getStyle()` und `setStyle()`. Diese beiden Methoden werden von allen Flex-Komponenten unterstützt, die von der UIComponent-Klasse erben. Die Methoden erlauben auch eine Änderung von Style-Eigenschaften von Komponenten während der Laufzeit. Der Nachteil dieser Vorgehensweise ist eine vergleichsweise hohe Prozessorlast, da die Flex-Anwendung bei jeder Styleänderung neu gerendert werden muss. Listing 4.20 zeigt das Beispiel für die Nutzung von `setStyle()`. Hier wird beim Event creationComplete die Schriftgröße und -farbe des Buttons mit der ID `myButton` festgelegt. Die gerenderte SWF zeigt das gleiche visuelle Ergebnis wie von Listing 4.17 (Abbildung 4.6).

Inline-Styles

Die Verwendung der Style-Eigenschaften von Flex-Komponenten stellt einen weiteren Weg dar, visuelle Anpassungen vorzunehmen. Die definierten Styles gelten dabei nur auf dieser Instanz der Komponente. Wie in Beispiel 4.21 zu sehen ist, ist diese Variante sehr effizient, da kein `<mx:Script>`-Block oder ActionScript-Methoden benötigt werden. Das visuelle Ergebnis ist auch hier das gleiche wie bei Beispiel 4.19 und 4.20.

27 vgl. StyleMangager-Klasse; Programmieren mit Flex 2, S. 354

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
    creationComplete="initApp()">
    <mx:Script>
        <![CDATA[
            public function initApp():void {
                myButton.setStyle("fontSize",15);
                myButton.setStyle("color",0xFF0000);
            }
        ]]>
    </mx:Script>
    <mx:Button id="myButton" label="Click Me"/>
    <mx:Button id="myButton2" label="Click Me"/>
</mx:Application>
```

Listing 4.20: Style-Definition mit Hilfe der setStyle()-Methode

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
    <mx:Button id="myButton" color="0xFF0000" fontSize="15"
        label="Click Me"/>
    <mx:Button id="myOtherButton" label="Click Me"/>
</mx:Application>
```

Listing 4.21: Beispiel für Inline-Style-Definition

4.4.6 Skins

Neben Stylesheets ist die Verwendung von Skins eine weitere Möglichkeit das visuelle Erscheinungsbild von Flex-Komponenten zu beeinflussen. Skinning beschreibt den Prozess des Austauschens von visuellen Elementen einer Komponente. Diese Elemente können Bitmaps, SWF-Dateien oder auch ActionScript-Klassen, die Methoden zum Zeichnen der Komponente enthalten. Ein Skin kann dabei eine ganze Komponente, teile einer Komponente oder auch nur das visuelle Erscheinungsbild einer Komponente in einem bestimmten Zustand definieren.

Auf der offiziellen Flex 3 Website[28] demonstriert Adobe das Prinzip von Skins anhand der Buttons-Komponente. Die Button-Komponente besitzt acht verschiedene Zustände (siehe Tabelle 4.4), zu denen jeweils ein andere Skin zugewiesen werden kann. Zu beachten ist, dass jeder Flex-Komponente eine Default-Skin-Klasse zugewiesen ist. So ist für die Gestalt eines Flex-Buttons standardmäßig die Klasse `mx.skins.halo.ButtonSkin` zuständig, die den Button in jedem Zustand rendert. Mit Hilfe der Skin-Eigenschaften des Buttons, können andere Skins zugewiesen werden.

28 vgl. http://livedocs.adobe.com/flex/3/html/help.html?content=styles_03.html#190648

State	Skin-Eigenschaft	Default-Skin-Klasse
down	downSkin	mx.skins.halo.ButtonSkin
over	overSkin	mx.skins.halo.ButtonSkin
up	upSkin	mx.skins.halo.ButtonSkin
disabled	disabledSkin	mx.skins.halo.ButtonSkin
selectedDisabled	selectedDisabledSkin	mx.skins.halo.ButtonSkin
selectedDown	selectedDownSkin	mx.skins.halo.ButtonSkin
selectedOver	selectedOverSkin	mx.skins.halo.ButtonSkin
selectedUp	selectedUpSkin	mx.skins.halo.ButtonSkin

Tabelle 4.4: Button-States mit den dazugehörigen Skin-Eigenschaften und Standard-Skin-Klassen

Im Folgenden wird auf drei verschiedene Typen von Skins eingegangen.

Grafikbasierte Skins

Die grafikbasierenden Skins sind gegenüber den anderen Skin-Typen sehr einfach zu erstellen. Bei dieser Art von Skins müssen lediglich Grafiken angegeben werden, die in den entsprechenden Zuständen der Komponente angezeigt werden sollen. Flex 3 unterstützt bei grafikbasierenden Skins Vektorgrafiken und Bitmaps, wie beispielsweise JPG-, GIF- und PNG-Dateien[29]. Auch Symbole, die in einer SWF eingebettet sind, können für einen Skin genutzt werden.

Programmatische Skins

Programmatische Skins sind vektorbasiert und bestehen aus einer Reihe von Linien-, Flächen- und Farbdefinitionen[30]. Mit Hilfe dieser Daten ist der Flash Player fähig, die Komponente zu zeichnen. Wird die Komponente skaliert, rotiert oder anderweitig modifiziert, kann sie vom Flash Player ohne erhöhten Aufwand neu gerendert werden. Weiterhin entstehen durch grafische Modifikationen an der Komponente kaum Qualitätsverluste, wie sie beispielsweise bei grafikbasierten Skins auftreten. Die programmatische Erzeugung des visuellen Erscheinungsbildes einer Komponente bringt noch weiter Vorteile gegenüber grafikbasierten Skins mit sich. So können beispielsweise runde Ecken für einen Button via ActionScript definiert werden, deren Radien mit der Skalierung der Komponente zunimmt. Ein weiterer Vorteil liegt im geringeren Speicherbedarf, da programmatisch erzeugte Skins keine Bitmap-Daten verwenden.

29 vgl. Skinning von Komponenten, Programmieren mit Flex 2, S. 368
30 vgl. Skinning von Komponenten, Programmieren mit Flex 2, S. 371

Zustandsabhängige Skins

Zustandsabhängige Skins[31] sind ein besonderer Typ von programmatischen Skins. Diese Art von Skins kann das Aussehen der Komponente in Abhängigkeit von dessen Zustand beeinflussen. Der Vorteil eines solchen Skins liegt darin, dass nur ein Skin für alle Zustände der Komponente benötigt wird, anstatt für jeden Zustand ein anderer Skin.

4.5 Beziehungen zwischen MXML und ActionScript

Wie schon oben erwähnt, sieht die Architektur von Flex vor, die visuelle Komponente der Anwendung mittels MXML umzusetzen und die Programmlogik mit ActionScript zu realisieren. So groß der Unterschied zwischen beiden Programmiersprachen auf dem ersten Blick auch erscheint — MXML und ActionScript 3 haben vieles gemeinsam. Bei der Kompilierung einer MXML-Datei wird in einem Zwischenschritt in reinen ActionScript 3 Code konvertiert. Auch der Aufbau einer MXML-Datei kann mit der Struktur einer ActionScript 3 Klasse verglichen werden.

Die Verwandtschaft von MXML und ActionScript 3 wird im Folgenden an der Instantiierung von Komponenten und der Parameterübergabe verdeutlicht.

4.5.1 Instantiierung von Komponenten

Flex stellt für die Instantiierung von MXML-Komponenten zwei Möglichkeiten zur Verfügung. Der Quelltext im Listing 4.22 zeigt ein Beispiel für die Erstellung einer Instanz eines <mx:Button> mittels MXML.

```
1 ...
2 <mx:Button id="button" />
3 ...
```

Listing 4.22: Flex-Button (MXML)

Der zweite Weg, eine <mx:Button>-Instanz zu erzeugen, bietet der Aufruf des Konstruktors der Buttonklasse mittels ActionScript 3. Der Quellcode von Listing 4.23 zeigt die Instantiierung eines <mx:Button> und die Zuweisung zur Variablen *button* mithilfe von ActionScript 3.

31 vgl. Creating stateful skins, Adobe Flex 3 Help, http://livedocs.adobe.com/flex/3/html/skinning_7.html#224961, aufgerufen am 7. März 2009

```
1 ...
2 var button:Button = new Button( );
3 ...
```

Listing 4.23: Flex-Button (ActionScript)

4.5.2 Parameterübergabe

Fast jede Flex-Komponente besitzt Eigenschaften, die mittels MXML-Tagattributen gesetzt werden können. Listing 4.24 zeigt ein Beispiel für die Beschriftung eines Button mithilfe des *label*-Attributs.

```
1 ...
2 <mx:Button id="button" label="Click" />
3 ...
```

Listing 4.24: Flex-Button mit Label (MXML)

Wurde eine MXML-Komponente durch ActionScript erstellt, so sind deren Attribute in den meisten Fällen durch Objektvariablen schreib- und lesbar. Listing 4.25 zeigt die gleiche Parameterübergabe wie in Listing 4.24 mittels ActionScript 3.

```
1 ...
2 var button:Button = new Button( );
3 button.label = "Click";
4 ...
```

Listing 4.25: Flex-Button mit Label (ActionScript)

Die beschriebenen Beispiele zeigen deutlich, dass eine MXML-Komponente grundsätzlich einer ActionScript 3 Klasse entsprecht und dass beide lediglich mit verschiedenen Syntaxen beschrieben werden. Selbst die MXML-Datei, die die Applikation beschreibt, ist eine ActionScript-Klasse, die von der Klasse `mx.core.Application` erbt. Auch selbst erstellte MXML-Komponenten sind Klassen, die von bestehenden Komponenten erben.

In Listing 4.26 ist der vollständige MXML-Quelltext einer einfachen Flex-Applikation zu sehen. Im Code ist gut zu erkennen, dass die Anwendung (so wie jede Flex-Anwendung) von der Klasse `mx.core.Application` erbt. Des Weiteren besitzt die Beispiel-Applikation die Eigenschaft *button1* vom Typ `mx.control.Button`.

Ähnlich wie in den Beispielen in den Unterabschnitten 4.5.1 und 4.5.2 kann auch diese Applikation mittels ActionScript 3 erstellt werden. Listing 4.27 zeigt eine ActionScript 3 Klasse, die die gleichen Eigenschaften wie das MXML-Pendant besitzt. Das Beispiel stellt dabei aber eine Vereinfachung einer tatsächlichen 1:1 Umsetzung in ActionScript 3 dar, um

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
    layout="absolute">
    <mx:Button id="button1" />
</mx:Application>
```

Listing 4.26: Einfache Flex-Applikation (MXML)

die grundsätzlichen Beziehungen von MXML und ActionScript 3 zu verdeutlichen. Für eine lauffähige Anwendung müsste erst die Initialisierung der Flex-Anwendung abgewartet werden, bis neue Komponenten per ActionScript erzeugt werden können.

```
package {
    import mx.core.Application;
    import mx.controls.Button;
    public class Example extends Application {
        internal var button1:Button;
        public function Example( ) {
            super( );
            button1 = new Button( );
            addChild(button1);
        }
    }
}
```

Listing 4.27: Einfache Flex-Applikation (ActionScript)

4.5.3 Inline-ActionScript

In Unterabschnitt 4.4.2 wird gezeigt, wie ActionScript 3 innerhalb einer MXML-Datei mit Hilfe des `Script`-Tags platziert werden kann. Diese Codezeilen verhalten sich wie Quellcode, der sich innerhalb einer ActionScript-Klasse befindet. Dementsprechend werden Variablendeklarationen innerhalb von MXML wie Membervariablen und Funktionsdefinitionen wie Methoden dieser Klasse behandelt. Das bedeutet zugleich, dass die Regeln, die beim Schreiben von ActionScript-Klassen befolgt werden müssen auch beim Schreiben von MXML-Komponenten eingehalten werden müssen. Die folgenden beiden Listings 4.28 und 4.29 zeigen die MXML- und ActionScript-Version einer Applikation, die die Variable *text* und die Methode `sayHello()` besitzt:

4.6 Adobe Flex Charting 2

Flex Charting ist eine kostenpflichtige ActionScript 3 Klassenbibliothek für Flex 2 und 3 von Adobe. Die Charting-Klassenbibliothek stellt zahlreiche Funktionen zur Verfügung, mit

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
    <mx:Script>
    <![CDATA[

        public var text:String;

        public function sayHello() : void {
        trace( "Hello!" );
        }
    ]]>
    </mx:Script>
    <mx:Button id="button1" />
</mx:Application>
```

Listing 4.28: Flex-Applikation mit Variablen- und Methodendeklaration (MXML)

```
package {
    import mx.core.Application;
    import mx.controls.Button;

    public class Example extends Application {
        public var text:String;

        public function Example( ) {
            super( );
        }

        public function sayHello() : void {
            trace( "Hello!" );
        }
    }
}
```

Listing 4.29: Flex-Applikation mit Variablen- und Methodendeklaration (ActionScript)

denen sich mit wenigen Codezeilen optisch ansprechende Diagramme erstellen lassen. Laut Adobe bietet Flex Charting 2 eine vielseitige Palette an Grafiken und Diagrammen für die Erstellung von Datenkonsolen und interaktiven Datenanalysen mit dem Flex 2-SDK[32]. Alle Charting Komponenten lassen sich, so wie die Standard-Komponenten des Flex 3 SDK, mittels CSS-Styles optisch anpassen.

Die vollständige API-Dokumentation und der mitgelieferte dokumentierte Quellcode ermöglichen eine nahtlose Einbindung von Flex-Charting-Komponenten in eine bestehende Flex-Anwendung. Die Flex-Charting-Erweiterung kommt in der in dieser Arbeit vorgestellten Beispielapplikation zur Visualisierung von Daten zum Einsatz. In Abbildung 4.8 ist ein Beispiel für eine `LineChart`- und eine `AreaChart`-Komponente zu sehen[33].

32 vgl. Adobe Flex Charting, Adobe: http://www.adobe.com/de/products/flex/charting/, aufgerufen am 7. März 2009

33 Screenshot entnommen aus dem *Adobe Flex-Charting Explorer*, http://examples.adobe.com/flex2/inproduct/sdk/explorer/explorer.html, aufgerufen am 7. März 2009

Abbildung 4.8: Beispiel einer `LineChart`- und `AreaChart`-Charting Komponente

4.7 Adobe Flex Builder 3

Adobes Flex Builder 3 ist eine kostenpflichtige Entwicklungsumgebung für Flex 3. Das Programm ist in englischer und japanischer Sprache für 200,- Euro (249,- US-Dollar) in der Standard Edition und für ca. 500,- Euro (699,- US-Dollar) in der Professional Edition erwerbbar. Die Standard Edition des Flex Builders 3 ist für Studenten auch kostenlos erhältlich.

Die Flex Builder Applikation wurde ursprünglich von Macromedia entwickelt[34]. Nach dessen Übernahme wird die Entwicklungsumgebung nun von Adobe weiterentwickelt und vorangetrieben. Die erste Version des Flex Builders konnte als Extension für den Macromedia Dreamweaver erworben werden und stellte somit kein eigenständiges Programm dar[35]. Viele Features, die Developer von einem Entwicklungsframework erwarten, haben in dieser Version noch gefehlt. Macromedia plante die nachfolgenden Versionen des Builders auf der weit verbreiteten Softwareplattform Eclipse zu basieren. Diesen Plan verfolgte Adobe konsequent weiter und veröffentlichte die Version 2 des Flex Builders als eigenständiges, aber auf Eclipse basierendes Programm. Wahlweise kann man den Flex Builder 2 auch als Plugin in eine bereits bestehende Eclipse-Installation integrieren.

4.7.1 Quellcode-Editor

Der Quellcode-Editor von Adobes Flex Builder richtet sich hauptsächlich an Programmierer. Der Editor bietet eine sehr fortschrittliche Programmierumgebung, die für MXML,

34 vgl. Flex Builder Feature Creeping, http://scalenine.com/blog/2007/03/02/flex-builder-feature-creeping/, aufgerufen am 7. März 2009

35 vgl. Adobe Flex Support Center, http://www.adobe.com/support/flex/downloads_updaters.html#flex15, aufgerufen am 7. März 2009

ActionScript, CSS und viele weitere Flex-spezifische Dateiformate optimiert ist.

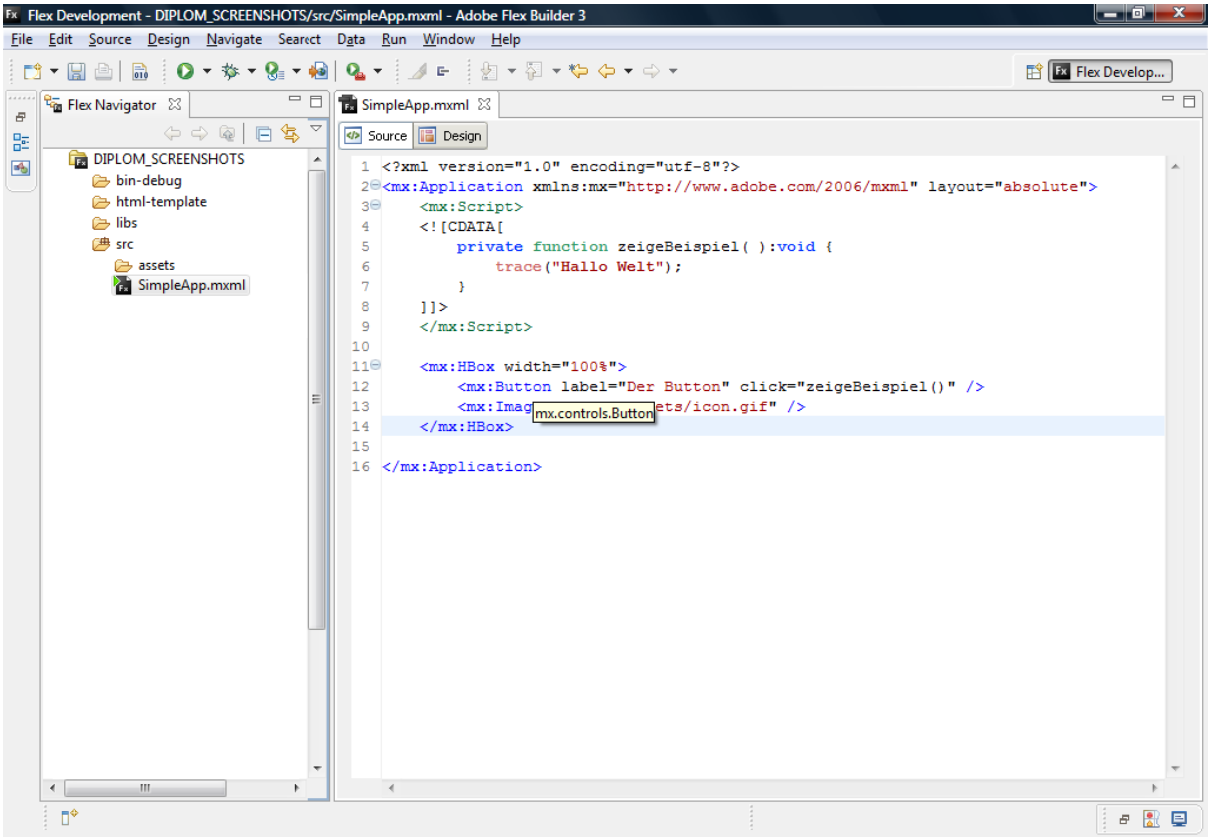


Abbildung 4.9: Sourcecode-Editor Ansicht des Flex Builder 3

Im Folgenden werden die herausragendsten Features des Flex Builders 3 erläutert.

Syntaxhervorhebung

Der Editor des Flex Builders hat die Fähigkeit, die Wörter und Zeichen des angezeigten MXML- und ActionScript-Codes in Abhängigkeit ihrer Bedeutungen in unterschiedlichen Farben, Schriftarten und -stilen darzustellen. Diese Funktion wird *Syntaxhervorhebung* bzw. im englischen *Syntax-Highlighting* genannt und gilt als eine der Standardfunktionalitäten heutiger integrierter Entwicklungsumgebungen.

Code-Formatter

Der vom Programmierer erstellte Quellcode kann mit Hilfe des im Flex Builders integrierten *Code-Formatters* (engl., Quellcode-Formatierer) in eine durch Regeln definierte Formatierung gebracht werden. Die Code-Formatierungsfunktionalität soll zusammen mit dem Syntax-Highlighting die Lesbarkeit des Quellcodes verbessern.

Code-Completion

Eines der wichtigsten Features des Flex Builder Editors ist die sogenannte Code-Completion (engl.: Quellcodevervollständigung). Die Code-Completion-Funktionalität des Flex Builders (Abb. 4.10) bietet dem Programmierer beim Schreiben Quellcode-Vorschläge in Form einer kleinen Liste an. Die angezeigten Vorschläge basieren auf der Flex-Quellcode-Bibliothek und auf den selbst erstellten MXML- und ActionScript-Klassen. Code-Completion beschleunigt nicht nur die Programmierung, sondern hilft dem Programmierer beim Schreiben von validem Code.

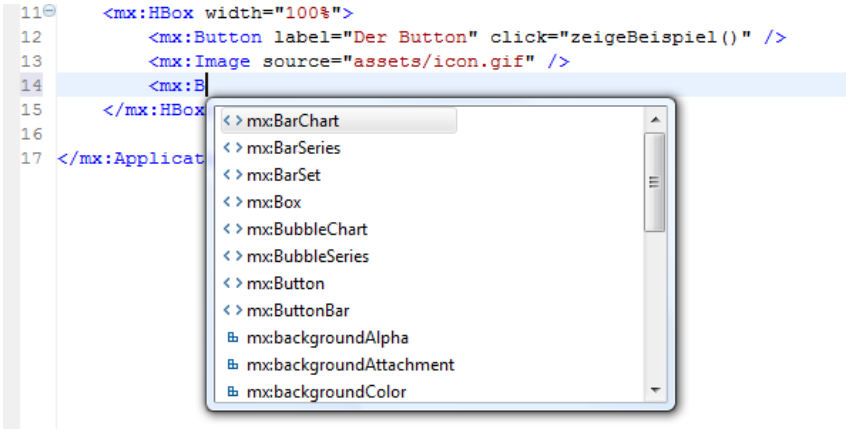


Abbildung 4.10: Aufgeklappte Liste mit Quellcode-Vorschlägen (Codecompletion), die vom Flex Builder 3 generiert wurden

4.7.2 Design-Werkzeug

Mit der Design-Ansicht, stellt der Flex Builder einen vollwertigen WYSIWYG-Editor zur Verfügung. Mit diesem Werkzeug lassen sich Benutzer-Oberflächen rein graphisch erstellen. Dieses Tools ist somit stark an Designer ohne Programmier-Fähigkeiten gerichtet. Alle proprietären UI-Komponenten können je nach Bedarf auf die Bühne gezogen werden. Alle Eigenschaften der Komponenten können mittels Eingabefeldern geändert werden. Bei jedem Hinzufügen von UI-Komponenten auf die Bühne wird automatisch der entsprechende MXML-Code generiert, der in der Quellcode-Editor-Ansicht wiederum betrachtet und geändert werden kann.

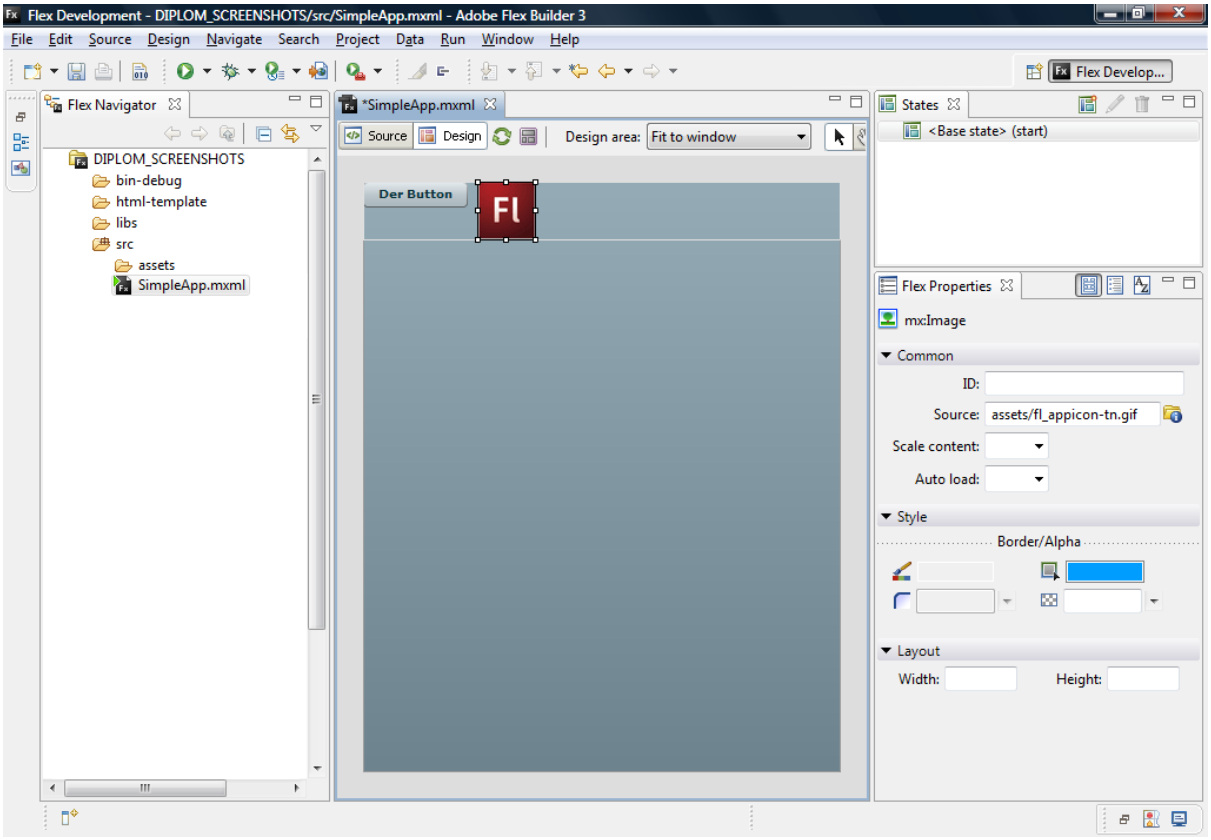


Abbildung 4.11: Design-Editor Ansicht des Flex Builder 3

4.8 Adobe Integrated Runtime

Adobe Integrated Runtime (AIR) ist eine Laufzeitumgebung und soll laut Adobe den Einsatz bewährter Web-Technologien für die Entwicklung plattformübergreifender Rich-Internet-Anwendungen für den Desktop ermöglichen[36]. Adobe begann Ende 2005 – direkt nach der Übernahme von Macromedia – mit der Entwicklung von Air unter dem Projektnamen „Apollo“. Am 25. Januar 2008 veröffentlichte Adobe zusammen mit Flex 3.0 schließlich die erste Version von Air.

Die neue Technologie soll die Vorteile der Internet-Technologien mit denen klassischer Desktop-Anwendungen kombinieren. So werden Technologien wie Flex/Flash, HTML/Ajax oder PDF von Air unterstützt, ohne dass die Anwendung in einem Browser laufen muss. Ermöglicht wird dies, durch eine eigens von Adobe entwickelte Laufzeitumgebung.

Ist die Runtime auf dem Rechner installiert, können RIA-Applikationen wie normale Anwendungen, beispielsweise vom Desktop oder aus dem Startmenü (bei Windows) heraus, aufgerufen werden. Des Weiteren stellt AIR erweiterte APIs zur Verfügung, die einen Zugriff

36 vgl. Adobe Air Website, www.adobe.com/de/products/air/, aufgerufen am 12. März 2009

auf das System gewährleisten. Unter anderem werden dadurch Drag-and-Drop und der Zugriff auf das Dateisystem möglich.

Die Laufzeitumgebung von Adobe AIR ist in der Version 1.5 kostenlos von der Hersteller-Website herunterladbar. Derzeit werden die Betriebssysteme Windows und Mac OS X unterstützt. Eine Unterstützung für Linux ist in Entwicklung und befindet sich noch im Alpha-Stadium.

4.9 Versionshistorie

Betrachtet man die Entwicklung von Flex kann man auf eine mehrjährige Versionshistorie zurück schauen. Die erste Version des Frameworks wurde bereits 2004 veröffentlicht. Aktuell steht Flex in Version 3.0 zum Download auf der Adobe-Website bereit. Zum Zeitpunkt der Erstellung dieser Arbeit ist bereits die nächste Flex-Version in Entwicklung[37]. Im Folgenden werden die einzelnen Versionen im Detail erläutert.

4.9.1 Version 1.0 und 1.5

Im März 2004 veröffentlichte Macromedia die erste Verkaufs-Version des bis dahin unter dem Codenamen Royale bekannte Präsentations-Server- und Applications-Framework von Flex. Bereits im Oktober des selben Jahres gab es ein Update auf Version 1.5. Macromedia bezeichnete den Zielmarkt für sein Produkt mit „enterprise application development“. Flex-Applikationen unterstützten Standards wie Scalable Vector Graphics (SVG), Cascading Style Sheets (CSS), Extensible Markup Language (XML), Web-Services auf Basis des Simple Object Access Protocol (SOAP) und ECMAScript. Der Preis für eine Serverlizenz lag bei 15.000 $/CPU – ein durchaus üblicher Preis für Serversoftware.

Das Flex 1.5-Paket umfasste einen J2EE Application-Server, der die von Macromedia entwickelte „Flex MarkUp Language“ (MXML) und ActionScript on-the-fly in Flash Applikationen kompilieren konnte. Des Weiteren stellt Macromedia vorgefertigte MXML-Code-Bausteine zur Verfügung, die zusammen mit der textbasierten und standardisierten Programmiermethodologie von Flex vor allem Entwickler von Business-Anwendungen ermöglichen sollten, existierende Werkzeuge und Designvorlagen sowie auf vorhandene Infrastruktur zurückgreifen zu können. Hinzu kamen Runtime-Services für die Anbindung von Datenbanken sowie das Management und die Verbreitung der Applikationen.

37 vgl. Gumbo, http://opensource.adobe.com/wiki/display/flexsdk/Gumbo, aufgerufen am 7. März 2009

Für die Programmierung von Flex-Anwendungen beinhaltete eine Serverlizenz fünf Lizenzen der Entwicklungsumgebung Flex Builder. Das auf Dreamweaver MX 2004 basierende Programmiertool stellte dem Programmierer Funktionen zum visuellen Layout sowie für die integrierte Entwicklung und Fehlersuche (Debugger) für Flex-Anwendungen zur Verfügung.

4.9.2 Adobe Flex 2.0

Flex 2.0 wurde am 27. Juni 2006 von Adobe veröffentlicht. Obwohl Adobe mit dieser Version einen kompletten Neuanfang wagte, dauerte die Entwicklung nur 18 Monate. Alle Flex-Klassen wurden komplett in ActionScript 3 geschrieben, das sich zu diesem Zeitpunkt selbst noch in der Entwicklung befand. Zum Abspielen der mit Flex erstellten SWF-Dateien wurde die Flash Player 9 Runtime benötigt. Zeitgleich mit Flex 2 wurde auch die Entwicklungsumgebung Flex Builder 2 veröffentlicht. Der Flex Builder 2 basiert nicht mehr auf dem Macromedia-Tool Dreamweaver, sondern wurde, auf Basis des Open-Source-Frameworks Eclipse, komplett neu entwickelt. Für die Kompilierung der Flex-Applikationen wurde ein vollkommen neues Konzept entwickelt. Wurde in den Vorversionen noch eine sehr kostenintensive Serversoftware benötigt, übernimmt nun die IDE bzw. der Kommandozeilen-Compiler des Flex SDKs die Kompilierung der Flex-Anwendung.

Adobe Flex 2.0.1

Anfang Januar 2007 gab es ein bedeutendes Update[38] des Flex Frameworks auf Version 2.0.1. Das Update beinhaltete vor allem ein Update des Flex Builder 2. Die Entwicklungsumgebung war nun neben Microsoft Windows auch zu Macintosh-Computern kompatibel. Des Weiteren unterstützt der Flex Builder 2 nun Eclipse 3.2. Bereits im August 2006 veröffentlichte Adobe das Programm ASDoc, das nun mit Version 2.0.1 in nochmals überarbeiteter Form fester Bestandteil des Flex-SDK wurde. Mit ASDoc ist es möglich eine Quellcode-Dokumentation aus den ActionScript- und MXML-Dateien generieren zu lassen. Mit dem Update wurden auch bedeutende Features dem Flex-SDK-Framework selbst hinzugefügt. So ist es mit Flex erstmals möglich CSS-Style-Sheet-Dateien nicht nur bei der Kompilierung der Anwendung einzubinden, sondern erst zur Laufzeit nachzuladen.

Eines des von den mittlerweile zahlreichen Flex-Entwicklern am meisten erhoffte Feature, ist die Möglichkeit, eine modular aufgebaute Flex-Anwendung zu programmieren. Diesen Wunsch ist Adobe nachgegangen[39] und fügte in Flex 2.0.1 das `mx.modules` Package hinzu. Das Flex

38 vgl. Introducing Flex 2.0.1, Adobe - Developer Center, http://www.adobe.com/devnet/logged_in/mchotin_flex201.html, aufgerufen am 7. März 2009

39 vgl. Modular Applications (Part 2), Adobe Blogs, http://blogs.adobe.com/rgonzalez/2006/06/modular_applications_part_2.html, aufgerufen am 7. März 2009

Framework unterstützte somit einen durchdachten Mechanismus, um große Anwendungen in kleinere Teile aufzuspalten.

4.9.3 Adobe Flex 3.0

Nach drei Beta Versionen von Flex 3 veröffentlichte Adobe schließlich am 26. Juni 2007 die finale Version 3 des Flex Frameworks. Gleichzeitig zu Flex 3 wurde die erste Version der *Adobe Integrated Runtime* (AIR) herausgebracht. Mit Flex 3 sind zahlreiche bedeutende Features hinzugekommen. Matt Chotin[40], einer der Produktmanager des Adobe Flex Teams, schildert die wichtigsten neuen Features von Flex 3. In den folgenden Punkten wird eine Zusammenfassung der Neuerungen aufgelistet:

Nativer Support für Adobe AIR

In Version 3 des Flex Frameworks wurde erstmals eine native Unterstützung von Adobe AIR integriert, indem Adobe AIR Entwicklungswerkzeuge direkt mit in das SDK aufgenommen wurden. Auch der Flex Builder, mittlerweile auch in Version 3, wurde mit AIR-Unterstützung ausgestattet.

Persistentes Framework-Caching

Mittels einer neuen Caching-Funktion ist es mit Flex 3 möglich Applikationen zu erstellen, die nur 50 KByte Speicher benötigen. Ohne diese Technik war es bis dahin nicht möglich eine Anwendung unter 200 KByte zu erstellen, da beim Kompilierungsvorgang das komplette Flex Framework eingebunden wird. Die Speichereinsparung wird erreicht, da der neu eingeführte Flash-Player eine Caching-Funktion für Adobe-eigene Komponenten besitzt.

Neue Flex Builder Tools

Im Flex Builder 3 wurden weitere Werkzeuge integriert, die die Produktivität der Software-Entwicklung erhöhen sollen. Unter anderem wurde eine Code-Refactoring-Unterstützung, neue Profiler für Performance- und Speicher-Optimierungen und ein Quellcode-Generator in die Entwicklungsumgebung integriert.

Integration in Creative Suite 3

Das neue Flex Component Kit für Flash CS3 erlaubt es jetzt Flex-Komponenten mit Flash CS3 zu erstellen und diese problemlos in eine Flex-Anwendung zu integrieren. Des Weiteren stehen dem Flex-Entwickler nun weitere Hilfsfunktionen für die Erstellung von Skins zur Verfügung.

40 vgl. What's new in Flex 3, http://www.adobe.com/devnet/flex/articles/flex3_whatsnew.html, aufgerufen am 7. März 2009

Advanced DataGrid

Das Advanced DataGrid ist eine neue Flex-Komponente, die die bereits aus Flex 2 bekannte DataGrid-Komponente erweitert. Hinzugekommen ist eine Unterstützung zur Darstellung von hierarchischen Daten und eine Pivot-Tabellen-Funktionalität.

Erste Schritte zu OpenSource

Mit dem Release von Flex 3 wurden erste Schritte getätigt, Flex in ein OpenSource-Projekt zu überführen[41]. So wurde das Flex- und Flex Builder-Bugtracking-System[42] der Öffentlichkeit zugänglich gemacht. Außerdem wurden detaillierte Informationen über die weitere Entwicklung von Flex veröffentlicht.

4.9.4 Adobe Flex 4

Die Version 4 des Flex-Frameworks stand zum Zeitpunkt dieser Arbeit noch in der Entwicklungsphase. Das Framework mit dem Codenamen *Gumbo* soll im zweiten Halbjahr 2009 als finale Version zum Download bereit stehen. Doch schon jetzt ist es möglich Betaversionen der Software herunterzuladen und zu testen. Auf der offiziellen Website des Gumbo-Projekts[43] ist der Zeitplan für die Fertigstellung der Software dokumentiert. Außerdem werden hier die zahlreichen Neuerungen dargelegt, die in der finalen Version enthalten sein sollen. Einige dieser Features werden im Folgenden aufgelistet und erläutert.

Flash Player 10

Flex 4 Anwendungen benötigen die Flash Player 10 Laufzeitumgebung. Adobe listet die neuen Features des unter dem Codenamen „Astro“ entwickelten Players auf der Adobe Labs Website[44] auf. Dazu gehört unter anderem eine neu integrierte API für einfache 3D-Effekte, eine überarbeitete Textlayout-Engine, sowie hardwarebeschleunigtes Video- und Vektorrendering.

Neue ActionScript 3 Features

Wie Flex 3 basiert auch Flex 4 auf der Scriptsprache ActionScript 3. Allerdings wurde die Syntax von ActionScript 3 um zahlreiche Elemente erweitert und an die neuen Features des

41 vgl. Flex 3 wird Open Source, http://www.heise.de/newsticker/Flex-3-wird-Open-Source–/meldung/88875, aufgerufen am 7. März 2009

42 vgl Adobe Bug and Issue Management System, http://bugs.adobe.com/flex/, aufgerufen am 7. März 2009

43 vgl. Gumbo, http://opensource.adobe.com/wiki/display/flexsdk/Gumbo, aufgerufen am 7. März 2009

44 vgl Flash Player 10 Features, http://labs.adobe.com/technologies/flashplayer10/, aufgerufen am 7. März 2009

Flash Player 10 angepasst. Die Scriptsprache ist so vollständig kompatibel zu ECMAScript 4 und bietet vollständige Typisierung.

MXML 2009

Aufgrund der zahlreichen hinzugekommenen Features wurde der MXML Namespace von MXML 2006 auf MXML 2009 erneuert. Der neue Namespace beinhaltet neben den Gumbo-Features aber auch noch alle Definitionen von Flex 3. Eins der bedeutendsten Neuerungen ist das `<Library>` Tag, das wiederverwendbare grafische Objekte beinhaltet.

FXG 1.0

Flex 4 unterstützt das speziell für die Flash-Plattform neu entwickelte grafische Dateiaustauschformat FXG 1.0. Das XML-basierte Format enthält Low-Level-Daten von Grafik- und Textprimitiven. Weiterhin können Transformations- und Modifikationsparameter für Bitmaps als auch für Vektor-Shapes gespeichert werden. Das Modell der FXG-Daten ähnelt laut Adobe sehr der Rendering-Engine des Flash Player 10 und nutzt auch dessen volle grafischen Möglichkeiten. Des Weiteren wurde auf gute Erweiterungsmöglichkeiten des Dateiformats geachtet, um auch für zukünftige Flash-Player-Versionen gewappnet zu sein.

Two-Way Data-Binding

Im Gegensatz zu seinen Vorgängerversionen (siehe Unterabschnitt 4.4.3) unterstützt Flex 4 nun auch natives beidseitiges (two-way) Data-Binding. Waren bei Flex 3 noch zwei one-way Data-Bindings nötig, um ein two-way Data-Binding herzustellen, ist mit der erweiterten Syntax von Flex 4 nun nur noch eine Anweisung notwendig.

5 Beispiel-Applikation „Adgame-Statistik“

5.1 Ausgangssituation

Die Ausgangssituation von meinem Flex-Projekt war eine bereits bestehende Flash-Applikation von der Firma PLUSPOL interactive GbR. Die Anwendung erlaubt eine statistische Auswertung von Livedaten von dafür vorgesehenen Flash-Games.

Die Features der Applikation werden in Unterabschnitt 5.1.1 vorgestellt. Die technische Umsetzung sowie deren Probleme werden in Unterabschnitt 5.1.2 näher betrachtet.

5.1.1 Features

Die Applikation stellt die vorhandenen Statistik-Daten hauptsächlich in tabellarischer Form dar. Die Anzahl der Zugriffe und der Besucher werden mittels Balkendiagramm angezeigt. Einige Statistiken erlauben die Auswahl des Monats und des Jahres, von dem die Statistikdaten angezeigt werden sollen. Eine Beispielansicht der Applikation ist in Abbildung 5.1 zu sehen.

Folgende Statistikdaten konnten von der Anwendung dargestellt werden:

- ▷ Anzahl der Besucher pro Monat (Balkenstatistik)
- ▷ Anzahl der gespielter Spiele pro Monat (Balkenstatistik)
- ▷ Teilnehmer (Liste)
- ▷ Badwords (Liste)
- ▷ gesperrte Spieler (Liste)

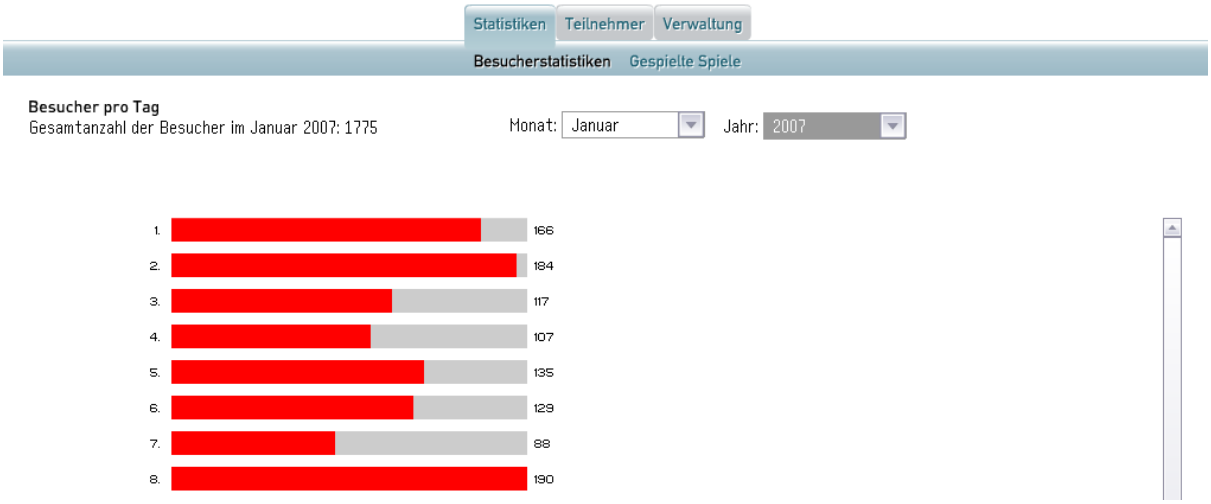


Abbildung 5.1: Screenshot der Vorgängerversion des in dieser Arbeit vorgestellten Statistikprogrammes. Anzeige der Besucherstatistik für Januar 2007

5.1.2 Technische Umsetzung

Das Programm wurde ausschließlich mit dem Flash 6 Authoring Tool und ActionScript 1 erstellt. Die Daten lädt das Programm per Flash-NetConnection als AMF-Datenstrom. In der Benutzeroberfläche kommen Flash-Komponenten, wie Checkboxen, Comboboxen, ScrollPanes und ScrollBars zum Einsatz. Die einzelnen Screens des Statistik-Tools sind fest implementiert und können ohne eine Neukompilierung nicht aktiviert beziehungsweise deaktiviert werden. Des Weiteren ist die komplette Anwendung in deutsch. Und auch hier ist die Integration einer neuen Sprache nur möglich, indem man die Sourcen überarbeiten und die SWF-Datei neu erstellen würde. Ein weiterer Nachteil in der technischen Umsetzung des Statistik-Programms liegt in der fehlenden Anpassungsmöglichkeit an verschiedene Adgame-Server. Wie schon bei den vorangehenden Problemen, muss auch hier eine Anpassung der Sourcen und Neukompilierung der Anwendung durchgeführt werden.

Da die Logik der Anwendung mit ActionScript 1 geschrieben wurde, ist auch keine echte Modularisierung der Applikation erkennbar. Das hat zur Folge, dass der Quellcode sehr unübersichtlich, für andere schwer nachvollziehbar und vor allem schlecht erweiterbar ist. Aber auch Anpassungen und Erweiterungen der Programmoberfläche sind nur relativ aufwendig und nur mit dem Flash Authoring Tool umsetzbar.

5.2 Zielstellung und Softwareanforderungen

Im Rahmen dieser Arbeit soll eine neue Flash-Applikation entwickelt werden, die eine qualitative und quantitative Verbesserung der statistischen Auswertung der Livedaten von Flash-Games – im Vergleich zu der bereits vorhandenen Anwendung – möglich ist. Die in Abschnitt 5.1 genannten Features der Vorgängerapplikation sollen dabei komplett übernommen und durch weitere nützliche Features ergänzt werden. Außerdem sollen bestehende Mängel und Unzulänglichkeiten des alten Statistikprogramms beseitigt werden.

Das Ziel ist in erster Linie eine intuitiv zu bedienende aber auch programmierfreundliche Anwendung mit reichhaltigen Funktionen zur Auswertung der Statistikdaten von Flash-Games. Im Folgenden werden die Key-Features der neuen Applikation erläutert.

Webbasierte Anwendung

Wie schon die Vorgängerversion, soll auch die neue Applikation vollständig webbasiert sein und mit einem Browser betrachtet werden können. Es ist vorgesehen, die Anwendung auf dem gleichen Server zu installieren, auf dem das Adgame selbst auch läuft. Es wird davon ausgegangen, dass der Nutzer gewillt ist eventuell benötigte Laufzeitumgebungen nachzuinstallieren.

Überarbeitete Benutzeroberfläche

Das Programm soll eine möglichst intuitiv zu bedienende Benutzeroberfläche (GUI) besitzen. Dazu gehören GUI-Elemente, wie sie bereits von Desktop-Anwendungen bekannt sind, wie beispielsweise Checkboxen, Comboboxen und scrollbare Listen. An die Gestaltung der Elemente existieren keine konkreten Vorstellungen, allerdings soll der visuelle Eindruck der Programmoberfläche zumindest aufgeräumt und übersichtlich wirken.

Erweiterte Visualisierung der Daten

Die in der Vorgängerversion verwendeten Balkendiagramme sollen durch eine verbesserte Variante ersetzt werden. Vor allem aber sollen mehrere Graphen in einem Diagramm darstellbar sein, um die Daten vergleichen zu können. Auch eine tabellarische Darstellung der Daten soll vorhanden sein.

Umfangreiche Konfigurationsmöglichkeiten

Jedes Adgame speichert unterschiedliche Statistikdaten. So muss beispielsweise in der Statistik eines Spiels, bei dem es keine Registrierung der Spieler gibt, auch keine Liste der registrierten Spieler geben. Aus diesem Grund ist es notwendig, dass das Funktionsspektrum des Statistikprogramms konfigurierbar ist, ohne das Programm neu kompilieren zu müssen.

Auch die Serververbindungsdaten, die bei jedem Spiel verschieden sind, sollen von „außen“ einstellbar sein.

Mehrsprachigkeit

Die Applikation soll Mehrsprachigkeit unterstützen, d. h. alle vorhandenen Texte und Beschriftungen müssen abänderbar sein, ohne das Programm neu kompilieren zu müssen.

Gute Erweiterbarkeit

Eines der Hauptprobleme der alten Anwendung war eine schlechte Erweiterbarkeit. Dies soll in der neuen Anwendung durch einen modularen Aufbau verbessert werden. Damit soll auch ein übersichtlicher Quellcode ermöglicht werden, so dass auch mit dem Projekt nicht vertraute Programmierer einen erleichterten Einstieg haben.

Vollständige Abwärtskompatibilität

Die neue Anwendung soll nicht nur für Statistiken zukünftiger Spiele nutzbar, sondern auch für die Darstellung der Daten alter Spiele einsetzbar sein. Dies macht vor allem eine Kompatibilität zum Adgame-Server notwendig.

Einfache Installation

Das Programm soll auf den Server installierbar sein, auf dem das Adgame selbst installiert ist. Da wegen eingeschränkten Zugriffsrechten eine Konfiguration des Webservers in vielen Fällen nicht möglich ist, darf die Installation des Statistikprogramms keine gesonderten Anforderungen an den Webserver stellen.

5.3 Technische Umsetzung

Die technische Umsetzung der Statistikapplikation muss unter Berücksichtigung der in Abschnitt 5.2 genannten Anforderungen erfolgen. Adobe Flex 3 wurde von vornherein vom Autor als Technologie für die technische Umsetzung gewählt. Eine genauere Betrachtung der Entwicklung des Statistikprogramms wird in den folgenden Unterabschnitten gegeben.

5.3.1 Flex-Applikation

Wie schon in Abschnitt 5.3 beschrieben, hat der Autor die Statistik-Applikation vollständig mit Flex 3 erstellt sowie jegliche Programmlogik in ActionScript 3 geschrieben. Flex 3 bietet sich besonders für die Umsetzung an, da das Flex 3 SDK kostenlos und gut dokumentiert ist.

Somit entfallen bei der Nutzung von Flex keine Lizenskosten und die Entwicklung wird durch die gute Dokumentation beschleunigt, wodurch nochmals die Entwicklungskosten gesenkt werden. Des Weiteren sind die mit Flex erstellten SWF-Dateien webbasierte Anwendungen, dessen Laufzeitumgebung Flash Player 9 bereits weit verbreitet ist.

Die Applikation ist in mehrere Module aufgeteilt, wodurch sich die Wartbarkeit und Erweiterbarkeit verbessert. Jedes Modul besteht aus einer MXML-Datei, die in die Main.mxml – der Hauptanwendung – eingebunden sind. Eine vollständige Liste der Module, deren Beschreibungen sowie Screenshots zu jedem Modul können in Abschnitt 5.5 betrachtet werden.

Da keine gesonderten Ansprüche an die visuelle Gestaltung der Anwendung existieren, werden keine globalen Styles oder gar selbst erstellte Skins benötigt. Somit kommt allein der Standard-Skin `mx.skins.halo` für die Darstellung der genutzten Komponenten zum Einsatz. Die Hauptnavigation besteht aus einer Tab-Navigation, die das Wechseln zwischen den Modulen ermöglicht. Die Applikation erlaubt eine beliebige Verschachtelung der Module in Tabs und Untertabs.

5.3.2 Serveranbindung

Die Kommunikation zwischen Flex-Anwendung und Server erfolgt über AMF 0. Die Funktionen, die der Server der Applikation anbietet, sind dabei die gleichen, die bei der Vorgängerversion genutzt worden. Diese Abwärtskompatibilität bringt den Vorteil, dass die bis dato existierenden Adgame-Server nicht für das neue Statistikprogramm aktualisiert werden müssen.

5.3.3 Konfiguration

Die anspruchsvollsten Anforderungen, die an die Statistik-Software gestellt werden sind die zahlreichen benötigten Konfigurationsmöglichkeiten. Dazu kommt noch, dass die Anzahl der möglichen Optionen zu Beginn der Entwicklung noch nicht feststand. Außerdem muss darauf geachtet werden, dass weitere Optionen für jede neue Komponente hinzukommen. Aus diesen Gründen hat sich der Autor für eine Konfiguration per XML-Datei entschieden, die bei jedem Programmstart geladen wird. Die Datei ist somit mit jedem Editor änderbar und besitzt eine leicht zu verstehende Struktur. Darüber hinaus können beliebige Optionen hinzugefügt oder geändert werden. Die XML-Struktur besteht aus mehreren Teilen, die im Folgenden näher betrachtet werden.

Applikationsname

Der im Tag <applicationname> angegebene Applikationsname wird in der Anwendung als Überschrift angezeigt.

Servereinstellungen

Die Optionen im Tag <server> lassen eine Konfiguration des Webservices zu, auf dem das Programm zugreift. Benötigt wird hier die URL zum AMF-Gateway und der Name der Serviceklasse.

Menüeinstellungen

Wie schon oben beschrieben wurde, ist die Tab-Navigation der Anwendung beliebig verschachtelbar. Nur die in diesen Einstellungen vorkommenden Programmmodule werden vom Programm angezeigt.

```
<menu>
  <item label="Besucher">
    <item label="Seitenaufrufe" module="modVisitsGame" />
    <item label="Gespielte Spiele" module="modPlayedGames" />
    <item label="Visits und Spiele" module="modVisitsPlayedGames" />
  </item>
  <item label="Spieler">
    <item label="Spielerliste Roboduell" module="modRoboPlayerList" />
    <item label="Spielerliste" module="modPlayerList" />
    <item label="Spieler eines Spiels" module="modPlayerListGame" />
    <item label="Spieler eines Tages" module="modPlayerListDaily" />
    <item label="Spieler eines Zeitraumes" module="modPlayerLRange" />
    <item label="Bundesländer" module="modPlayerState" />
  </item>
  <item label="Highscore" module="modHighscoreListRange" />
  <item label="Blacklisting">
    <item label="Badword-Liste" module="modBadwordList" />
    <item label="Gesperrte Accounts" module="modHiddenEntriesList" />
  </item>
  <item label="Gewinnerliste" module="modWinnerList" />
  <item label="Server Infos" module="modServerStat" />
</menu>
```

Listing 5.1: Beispiel für die Konfiguration der Tab-Navigation

Moduleinstellungen

Die Moduleinstellungen, die im Tag <modules> vorgenommen werden können, bieten die umfangreichsten Einstellungsmöglichkeiten. Unter anderem kann hier entschieden werden, welche Buttons und Funktionen für den Nutzer sichtbar beziehungsweise benutzbar sind. Auch die Daten, die vom Statistikprogramm angezeigt werden sollen, können hier eingestellt werden. In Listing 5.2 wird eine Beispielkonfiguration des Moduls *HighscoreListRange* gezeigt.

```
<module id="modHighscoreListRange" name="HighscoreListRange">
  <export>
    <url>export_csv.php</url>
  </export>
  <functions>
    <item id="buttonAddWinner" label="Gewinner hinzufügen" />
    <item id="buttonHidePlayer" label="Spieler sperren" />
    <item id="buttonPlayerDetails" label="Spielerdetails" />
    <item id="buttonExport" label="Exportieren" />
  </functions>
  <list>
    <col headerText="Member ID" dataField="id_member" />
    <col headerText="Login-Name" dataField="login" />
    <col headerText="Datum" dataField="date" />
    <col headerText="Score" dataField="score" dataType="Number" />
    <col headerText="Teiln. bed. bestätigt" dataField="confirm" />
    <col headerText="IP" dataField="ip" />
    <col headerText="Spieldatum" dataField="ts" />
    <col headerText="Name" dataField="name" />
    <col headerText="Vorname" dataField="first" />
    <col headerText="Email Adresse" dataField="email" />
  </list>
  <form id="playerDetails" labelDataField="login">
    <item dataField="id_member">Member-ID</item>
    <item dataField="login">Login</item>
    <item dataField="email">E-Mail</item>
    <item dataField="confirm">Teilnahmebed. bestätigt</item>
    <item dataField="newsletter">Newsletter abonniert</item>
    <item dataField="ip">IP Nummer</item>
    <item dataField="ts">Spiel-Datum</item>
  </form>
</module>
```

Listing 5.2: Beispiel für die Konfiguration des Moduls HighscoreListRange

Spracheinstellungen

Der Tag `<captions>` beinhaltet alle in der Applikation vorkommenden Texte. Somit kann eine vollständige Anpassung der Anwendung an eine beliebige Sprache vorgenommen werden.

```
<captions>
  <BUTTON_OK>OK</BUTTON_OK>
  <BUTTON_CANCEL>Abbrechen</BUTTON_CANCEL>
  <BUTTON_SAVE>Speichern</BUTTON_SAVE>
  <BUTTON_YES>Ja</BUTTON_YES>
  <BUTTON_NO>Nein</BUTTON_NO>
  <BUTTON_LOAD>Laden</BUTTON_LOAD>
  <BUTTON_LOAD_STATISTIC>Statistik laden</BUTTON_LOAD_STATISTIC>
  <BUTTON_EXPORT>Export</BUTTON_EXPORT>
  <TEXT_BACK>zurück</TEXT_BACK>
  <TEXT_NEXT>vor</TEXT_NEXT>
  <TEXT_BACK10>-10</TEXT_BACK10>
  ...
</captions>
```

Listing 5.3: Beispiel für die Definition der im Programm vorkommenden Texte

Debug-Modus

Die Debug-Einstellungen umfassen nur eine Option, die auf `true` oder `false` gesetzt werden kann. Wird das Debugging aktiviert, zeigt die Applikation ein zusätzliches Textfeld mit Debuginformation im unteren Bereich der Anwendung an.

5.3.4 Installation

So wie es in den Software-Anforderungen festgelegt ist, soll die Installation der Adgame-Statistik sehr einfach durchführbar sein und keine gesonderten Anforderungen an den Server stellen. Tatsächlich lässt sich die Software denkbar einfach auf den Server installieren. Dazu sind lediglich die folgenden Dateien auf den Webserver zu übertragen:

AdgameStat.swf ist die compilierte Statistik-Applikation. Sie beinhaltet alle Module, die je nach Konfiguration aktiviert bzw. deaktiviert werden können.

bundeslaender.swf wird von der Applikation benötigt, wenn das Modul *PlayerState* in der Konfiguration aktiviert worden ist.

index.html wird vom Browser aufgerufen und hat den Flashfilm *AdgameStat.swf* eingebettet. Die SWF-Datei kann auch in einer beliebigen anderen HTML-Seite eingebunden werden.

ags_config.xml beinhaltet die komplette XML-Konfiguration des Statistikprogramms.

swfobject.js enthält JavaScript-Funktionen die für die Einbettung der SWF-Datei in die *index.html* benötigt werden.

5.4 Entwicklungsumgebung

Die eingesetzte Software zum Entwickeln der Anwendung sollte so kostengünstig wie möglich, wenn nicht sogar kostenlos, sein. Aus diesem Grund hat sich der Autor bis auf eine Ausnahme, für Open Source Software entschieden. Die eingesetzte Software wird im folgenden kurz aufgelistet.

Adobe Flex 3 SDK. Das Flex 3 Framework ist schon ab Version 2 als Open Source verfügbar. Aus dem SDK werden die Flex-Bibliotheken und der Flex-Compiler für diese Projekt benötigt.

Adobe Flex Builder 3. Die Flex-Entwicklungsumgebung wurde vom Autor zu Beginn der Entwicklung als 30-Tage-Trial verwendet. Dank des Syntax-Highlightings und der hervorragenden Code-Completion-Funktionalität half es dem Autor einen leichteren Einstieg in die MXML- und ActionScript 3 Programmierung zu erhalten.

Adobe Flex Charting. Die Code-Bibliothek wurde in diesem Projekt für die Erstellung aller grafischen Diagramme benötigt. Sie ist das einzige kostenpflichtige Software, die in diesem Projekt zum Einsatz kommt.

Flash Develop 3. Das Open Source Projekt FlashDevelop 3[1] löste den Flex Builder 3 nach Ablauf der Testversion als Entwicklungsumgebung für dieses Projekt ab.

5.5 Programm-Module

Im Folgenden werden die einzelnen Programmmodule der Statistik-Anwendung beschrieben. Dabei wird kurz erklärt welche Daten angezeigt werden und welche Funktionen dem Nutzer zur Verfügung stehen, um beispielsweise nur bestimmte Daten vom Server zu Laden oder zu ändern. Zusätzlich wird zu jedem Modul ein Screenshot gezeigt.

Bei jedem Modul, das Informationen in Tabellenform darstellt, ist es möglich, diese Daten zu sortieren, indem man auf die Kopfzeile der zu sortierenden Spalte klickt. Einige spielerspezifische Informationen wurden aus datenschutzrechtlichen Gründen unkenntlich gemacht.

1 Flash Develop, http://www.flashdevelop.org/, aufgerufen am 12. März 2009

5.5.1 Modul VisitsGame

Das Modul *VisitsGame* ist eines der grundlegendsten Funktionen des Programms. Hier kann eine Statistik über die Anzahl der Besucher die das Spiel gestartet haben bzw. die Website, auf der das Spiel eingebunden ist, betrachtet haben. Der Nutzer des Statistikprogramms hat dabei die Möglichkeit einen bestimmten Zeitraum zu wählen in dem die tägliche Besucherzahl angezeigt werden soll. Die Ergebnisse können in Listenform und als Diagramm betrachtet werden. Sind mehrere Spiele oder Spiel-Modi vorhanden, können diese in einer Liste an- oder abgewählt werden. In der Listendarstellung der Ergebnisse werden die gewählten Spiele als separate Spalten oder in der graphischen Auswertung als anders farbige Kurve dargestellt.

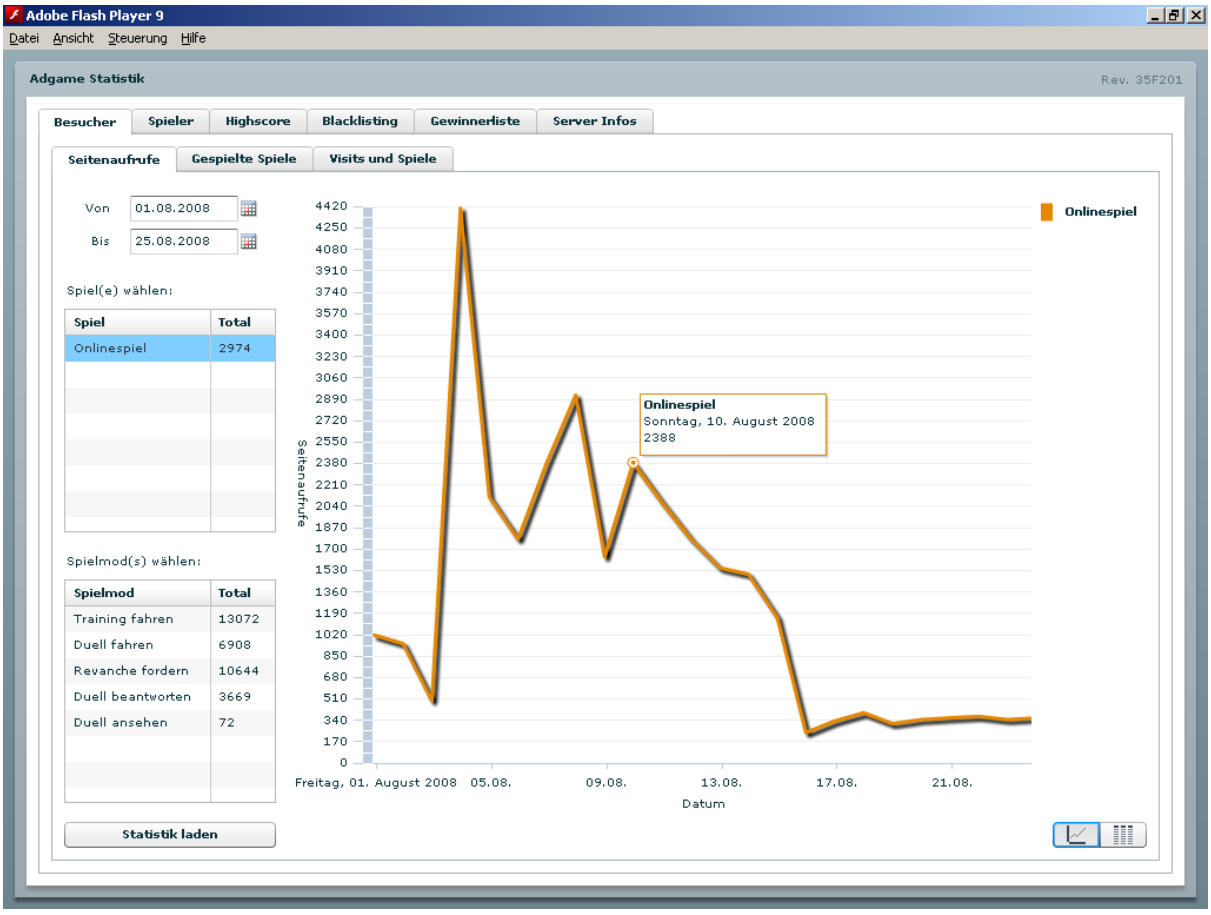


Abbildung 5.2: Grafische Auswertung der Besucherstatistik

5.5.2 Modul PlayedGames

Bei jedem Start des Spiels durch einen Spieler, wird ein Besuch auf dem Server registriert. Während eines Besuches kann ein Spieler (bei den meisten Adgames) beliebig viele Spiele spielen. Jedes dieser gespielten Spiele werden vom Server registriert *WinnerList* und gespeichert. Diese Daten können im Modul *PlayedGames* betrachtet werden. Das Modul ähnelt in der Bedienung und Funktionalität sehr dem *VisitsGame*-Modul. Der Nutzer kann einen Zeitraum festlegen, Spiele und Spiel-Modi wählen und die Ergebnisse als Liste oder auch als Diagramm betrachten.

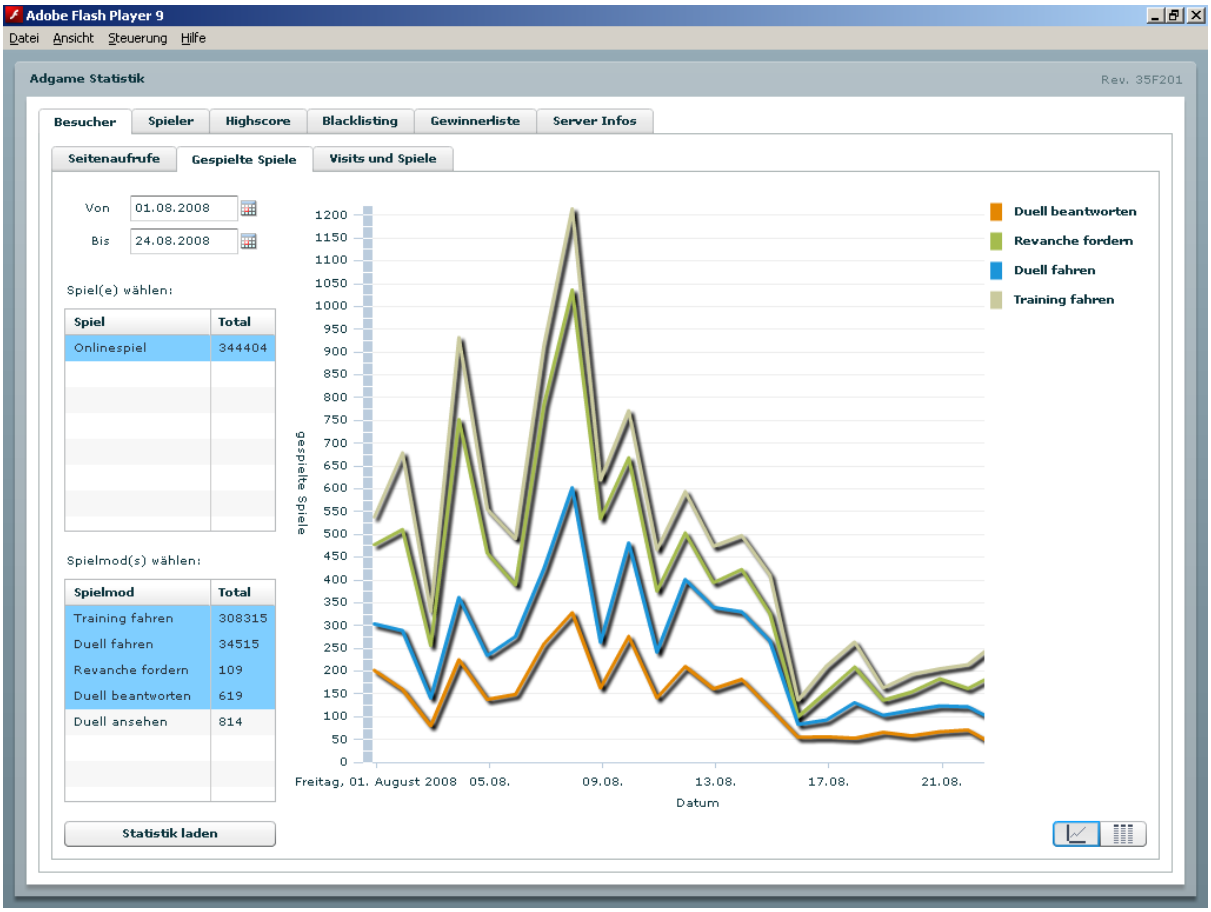


Abbildung 5.3: Grafische Auswertung der gespielten Spiele

5.5.3 Modul VisitsPlayedGames

Um eine Aussage über die Beliebtheit eines Spieles zu erhalten, kann es sehr nützlich sein, zu wissen, ob die Spieler eines Spiels nicht nur einmal, sondern mehrmals das selbe Spiel gespielt haben. Diese Informationen versucht das Modul *VisitsPlayedGames* darzustellen, indem es die gespielten Spiele und die Anzahl der Besuche des Spiels in ein und demselben Diagramm anzeigt. Die Daten stehen auch hier wie bei den Modulen *VisitsGame* und *PlayedGames* sowohl als Diagramm als auch in Tabellenform zur Verfügung.

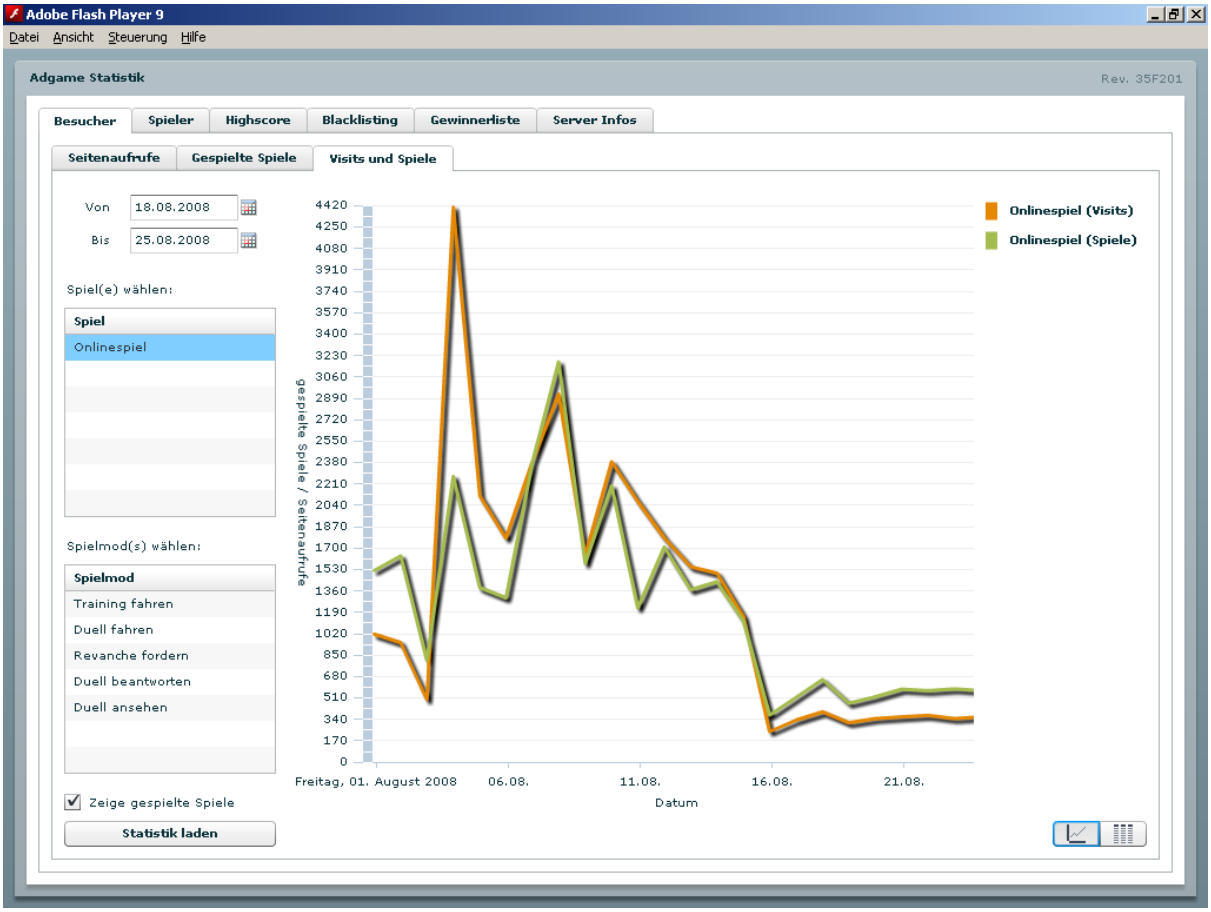


Abbildung 5.4: Gespielte Spiele und Visits

5.5.4 Modul BadwordList

Leider steht es an der Tagesordnung, dass sich viele Spieler von Adgames unschöne bzw. unerwünschte Nicknamen geben. Um einen möglichen Highscore-Eintrag oder ein Anlegen eines Profils mit einem solchen „hässlichen“ Namen zu verhindern, wird eine Liste mit verbotenen Begriffen – den sogenannten Badwords – geführt. Diese Liste kann mit dem Modul *BadwordList* betrachtet werden. Des Weiteren ist es mögliche vorhandene Badwords aus der Liste zu löschen aber auch Neue hinzuzufügen. Bei letzterer Aktion, durchsucht der Server alle vorhanden Spielernamen und Highscore-Einträge auf das hinzugefügte Badword und sperrt die gefundenen Einträge.

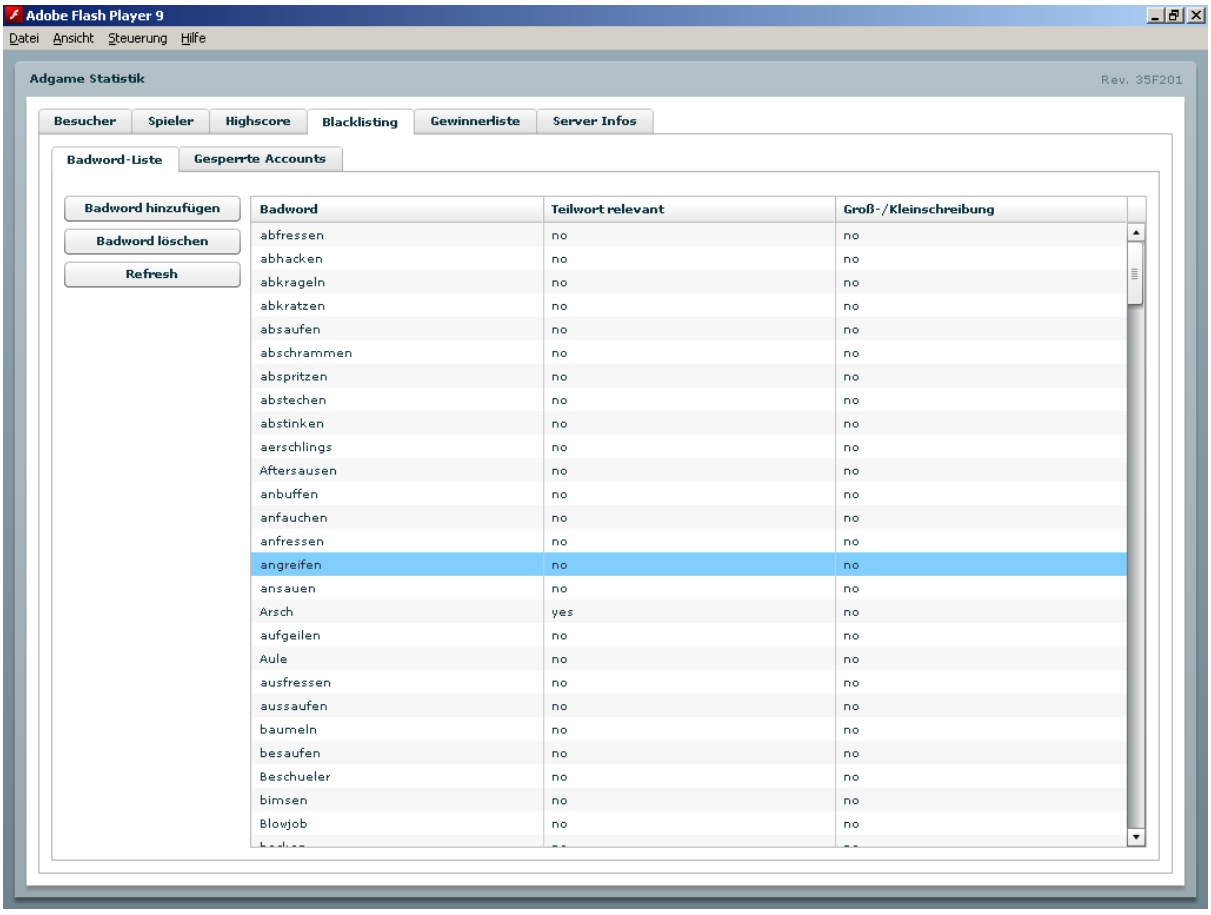


Abbildung 5.5: Beispiel einer Badword-Liste

5.5.5 Modul HiddenEntriesList

Alle Spieler-Profile oder -Namen, die anhand der Badword-Liste vom Server deaktiviert wurden, können im Modul *HiddenEntriesList* eingesehen werden. Auch die vom Nutzer des Statistik-Programms „von Hand“ gesperrten Spieler sind hier zu sehen. Das Programm bietet hier die Möglichkeit Einträge, die zum Beispiel fälschlicherweise durch ein zu allgemeines Badword gesperrt wurden, wieder freizuschalten.

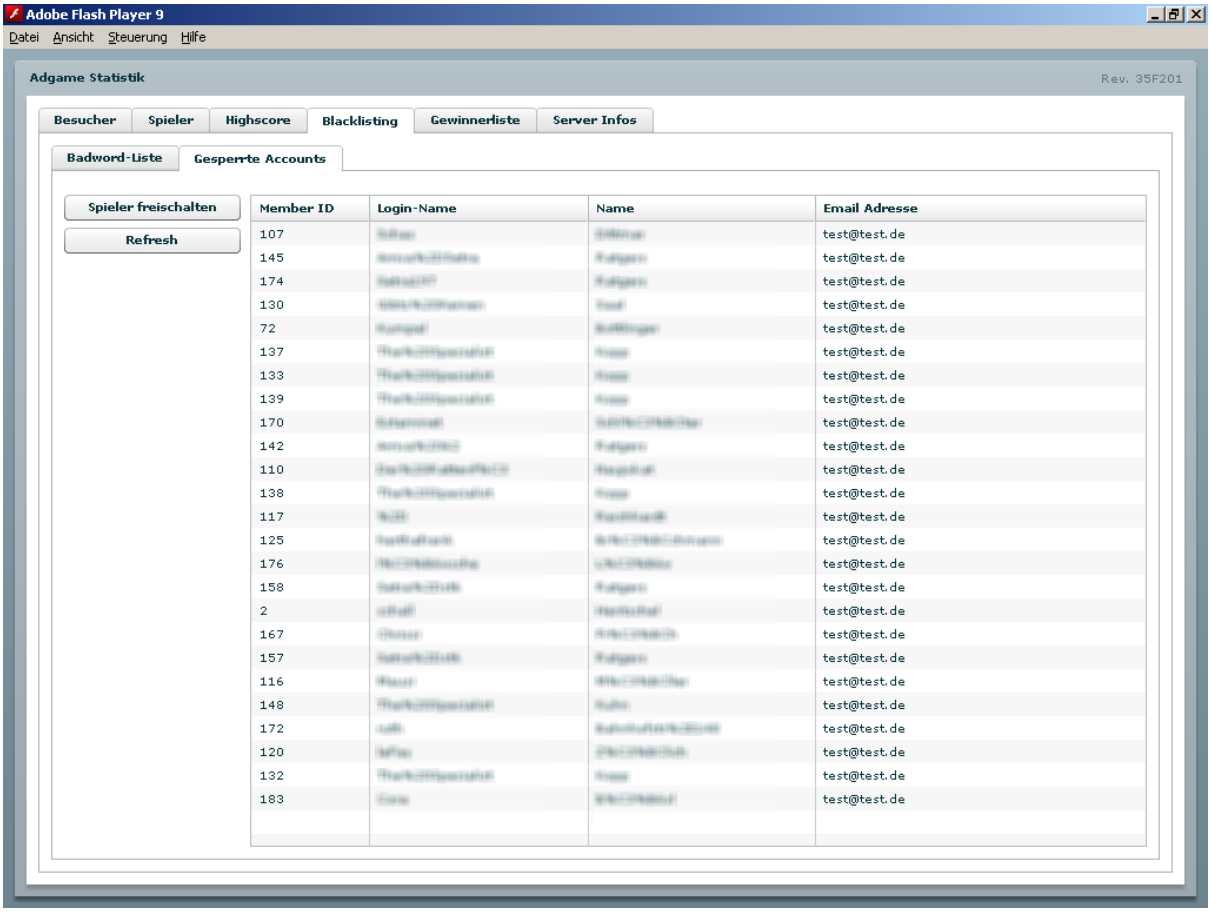


Abbildung 5.6: Liste mit gesperrten Spielern

5.5.6 Modul PlayerList

Das Modul PlayerList ist das umfangreichste Modul des Programms. Alle vorhandenen, nicht gesperrten Spieler werden hier in Listenform angezeigt. Dem Nutzer stehen dazu folgende Funktionen zur Verfügung:

Spielerdetails. Ist ein Spieler in der Liste markiert, kann diese Option gewählt werden. Dabei öffnet sich ein Fenster in dem die detaillierten Informationen des gewählten Spielers dargestellt werden.

Spieler sperren. Der oder die selektierte(n) Spieler werden gesperrt und können im Modul *HiddenEntriesList* betrachtet und auch wieder entsperrt werden. Ein gesperrter Spieler wird nicht mehr im Modul *PlayerList* angezeigt.

Gewinner hinzufügen. Der oder die selektierte(n) Spieler werden als Gewinner festgelegt. Dazu öffnet sich ein modales Fenster, um den Gewinner-Preis festzulegen. Wurde nur ein Spieler gewählt, können in diesem Fenster die Spielerdaten, wie Name und Wohnort, geändert werden. Wurden mehrere Spieler gewählt, so werden die Namen der Spieler nochmals in einer Liste dargestellt.

Refresh. Mit dieser Funktion wird die Spielerliste erneut vom Server geladen und angezeigt.

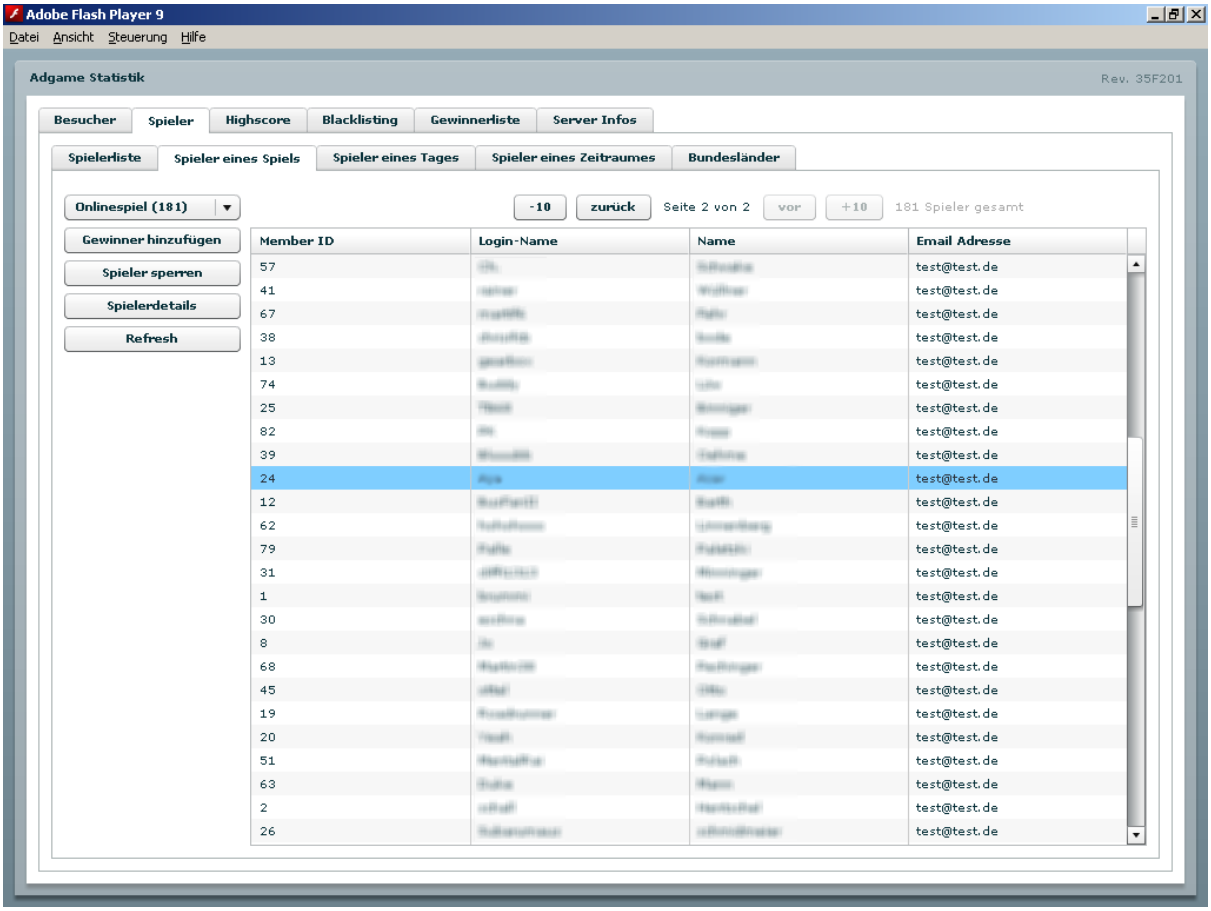


Abbildung 5.7: Liste aller registrierten Spieler

5.5.7 Modul PlayerListDaily

Dieses Modul stellt wie das Modul *PlayerList* Spieler in Listenform dar, die in der Datenbank vorhanden sind. Dabei sind auch hier die Funktionen „Spielerdetails“, „Spieler sperren“, „Gewinner hinzufügen“ und „Aktualisieren“ vorhanden. Der Unterschied zum Modul *PlayerList* liegt im Festlegen eines Datums, bevor die Suchanfrage getätigt werden soll. Als Ergebnis werden alle Spieler angezeigt, die an diesem Tag ein oder mehrere Spiele getätigt haben.

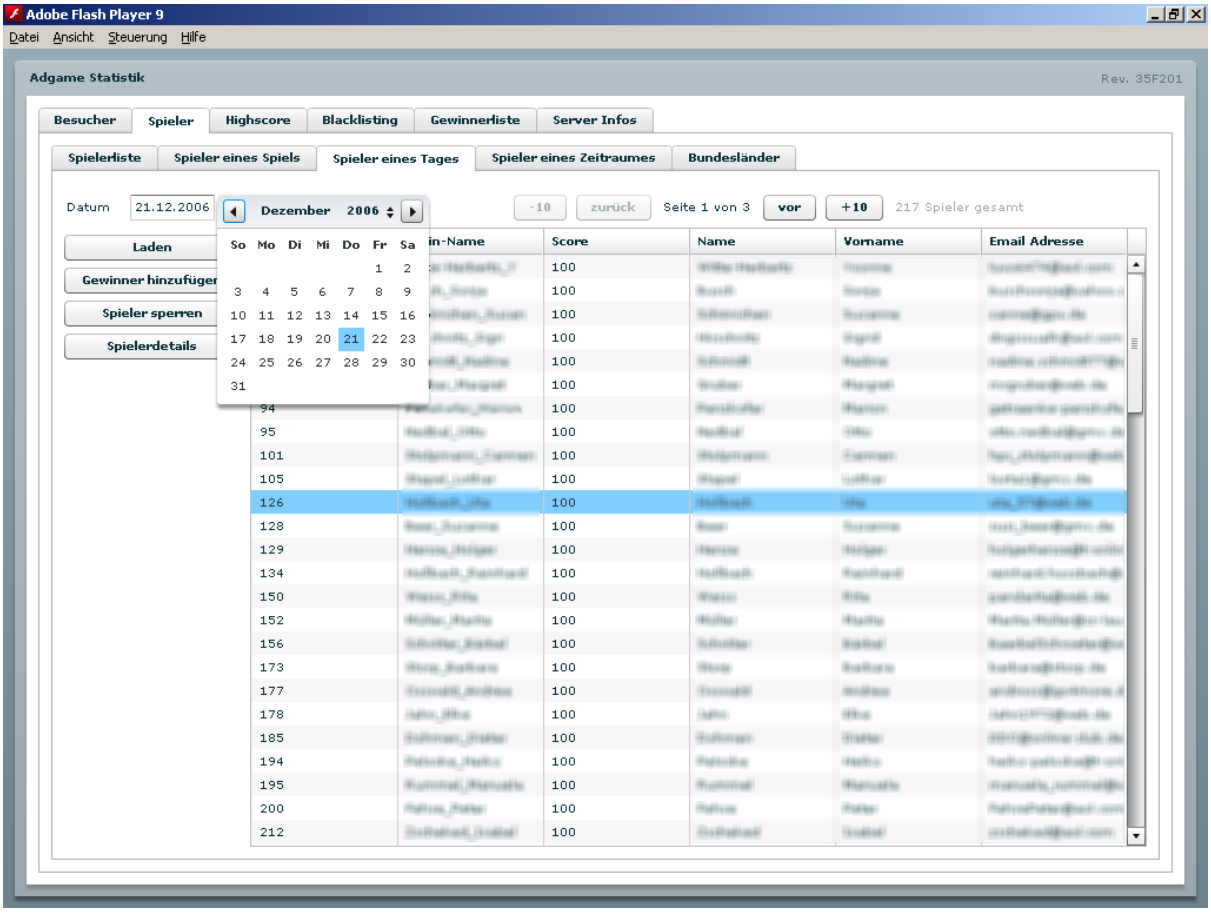


Abbildung 5.8: Liste von registrierten Spielern, die an einem bestimmten Tag gespielt haben

5.5.8 Modul PlayerListRange

Dieses Modul listet wie das Modul *PlayerList* Spieler auf, die in der Datenbank vorhanden sind. Dabei sind auch hier die Funktionen „Spielerdetails“, „Spieler sperren“, „Gewinner hinzufügen“ und „Aktualisieren“ vorhanden. Besonderheit dieses Moduls ist es, nach den Spielern zu suchen, die in einem bestimmten Zeitraum ein Spiel gespielt haben. Den gewünschten Zeitraumes kann man mittels zweier Datums-Eingabefeldern definieren.

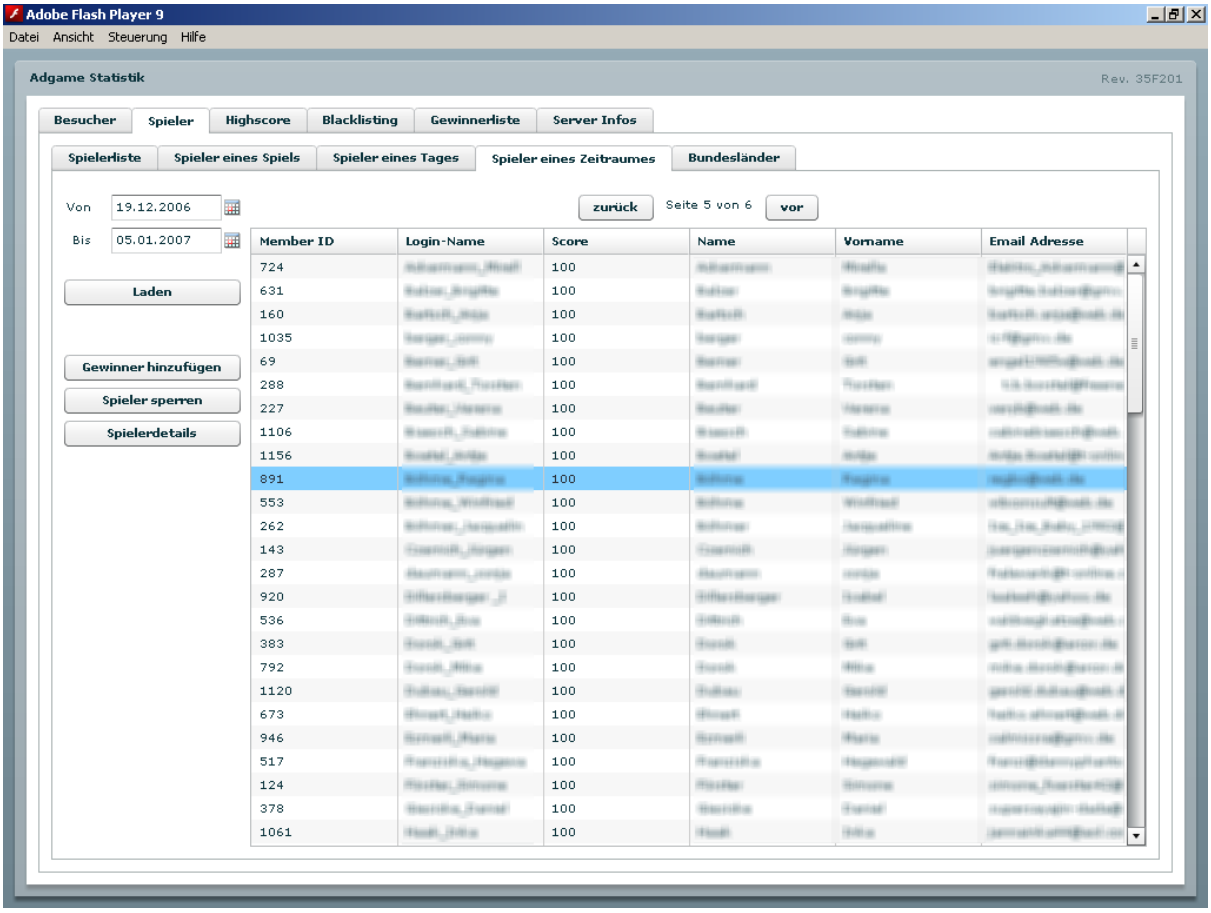


Abbildung 5.9: Liste von registrierten Spielern, die innerhalb des gewählten Zeitraumes gespielt haben

5.5.9 Modul PlayerListGame

Dieses Modul listet wie das Modul *PlayerList* Spieler auf, die in der Datenbank vorhanden sind. Dabei sind auch hier die Funktionen „Spielerdetails“, „Spieler sperren“, „Gewinner hinzufügen“ und „Aktualisieren“ vorhanden. Zusätzlich besteht noch die Möglichkeit nur die Spieler anzeigen zu lassen, die ein bestimmtes Spiel bzw. Spiel-Modus gespielt haben.

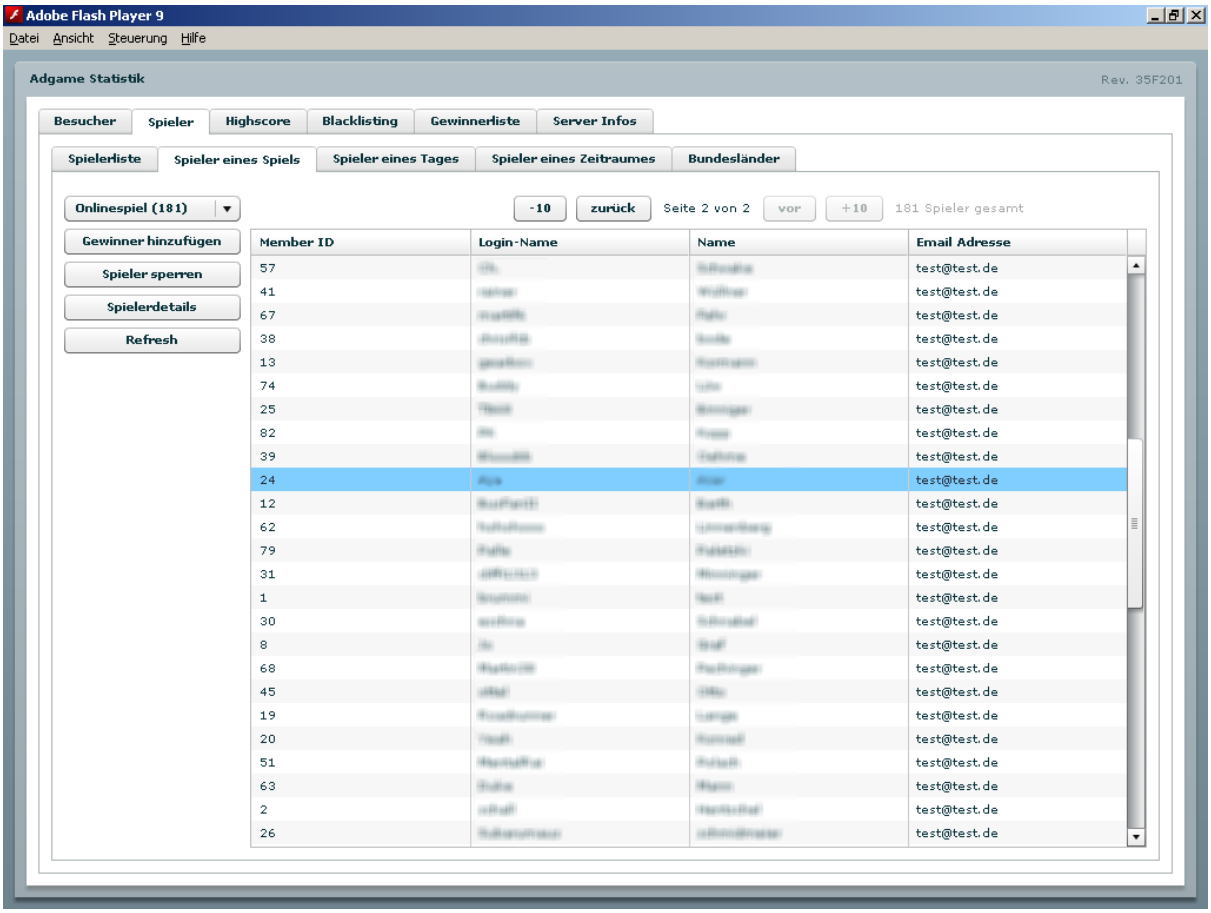


Abbildung 5.10: Liste von registrierten Spielern, die das gewählte Spiel gespielt haben

5.5.10 Modul HighscoreListRange

Eine wichtige Funktion des Statistik-Programms ist das Festlegen eines oder mehrerer Gewinner für ein Spiel. Im Allgemeinen wird dazu der Preis an denjenigen Spieler übergeben, der die höchste Punktzahl in diesem Spiel erreicht hat. Im Modul *HighscoreListRange* kann man dazu die Highscore-Liste einsehen und den oder die Gewinner anhand seiner Platzierung festlegen. Des Weiteren ist es möglich detaillierte Information über die Spieler zu erhalten oder auch Spieler zu sperren. Letztere Funktion ist in diesem Modul besonders interessant, da hier Spieler mit einer ungewöhnlich hohen Highscore zumeist Betrugsversuche darstellen und gesperrt werden können.

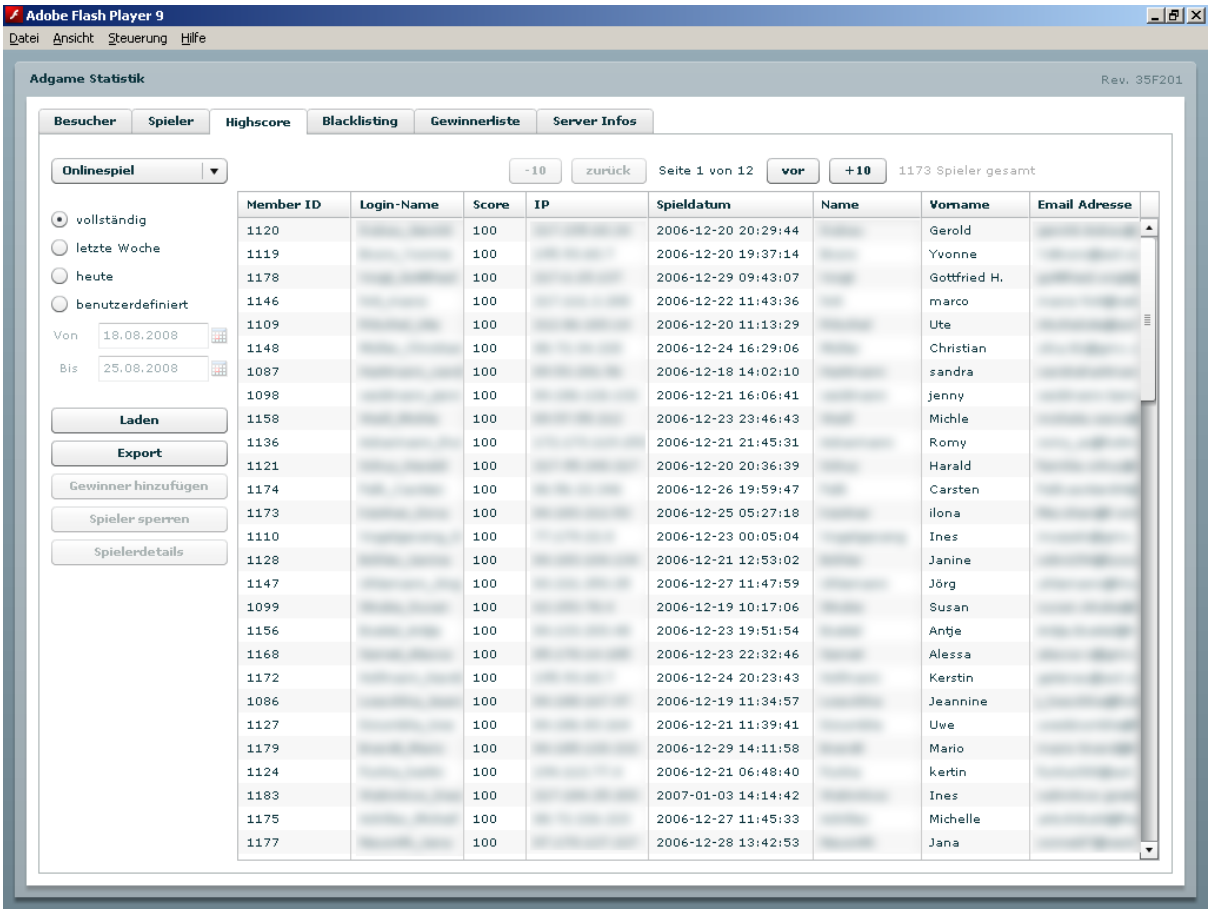


Abbildung 5.11: Highscoreliste mit Spielern, die innerhalb des gewählten Zeitraumes gespielt haben

5.5.11 Modul WinnerList

Wurden Gewinner für ein Spiel bestimmt, so werden diese im Modul *WinnerList* in einer Tabelle aufgelistet. Die Daten der Spieler in dieser Liste werden zumeist dazu verwendet, sie auf einer Website zu Veröffentlichen. In nicht wenigen Fällen ist es allerdings nötig, die vom Spieler erstellten Account-Informationen hinsichtlich Groß- und Kleinschreibung zu verändern oder Rechtschreibfehler zu korrigieren. Dazu bietet das Modul *WinnerList* die Funktion „Gewinner bearbeiten“. Des Weiteren können Gewinner aus der Liste gelöscht werden.

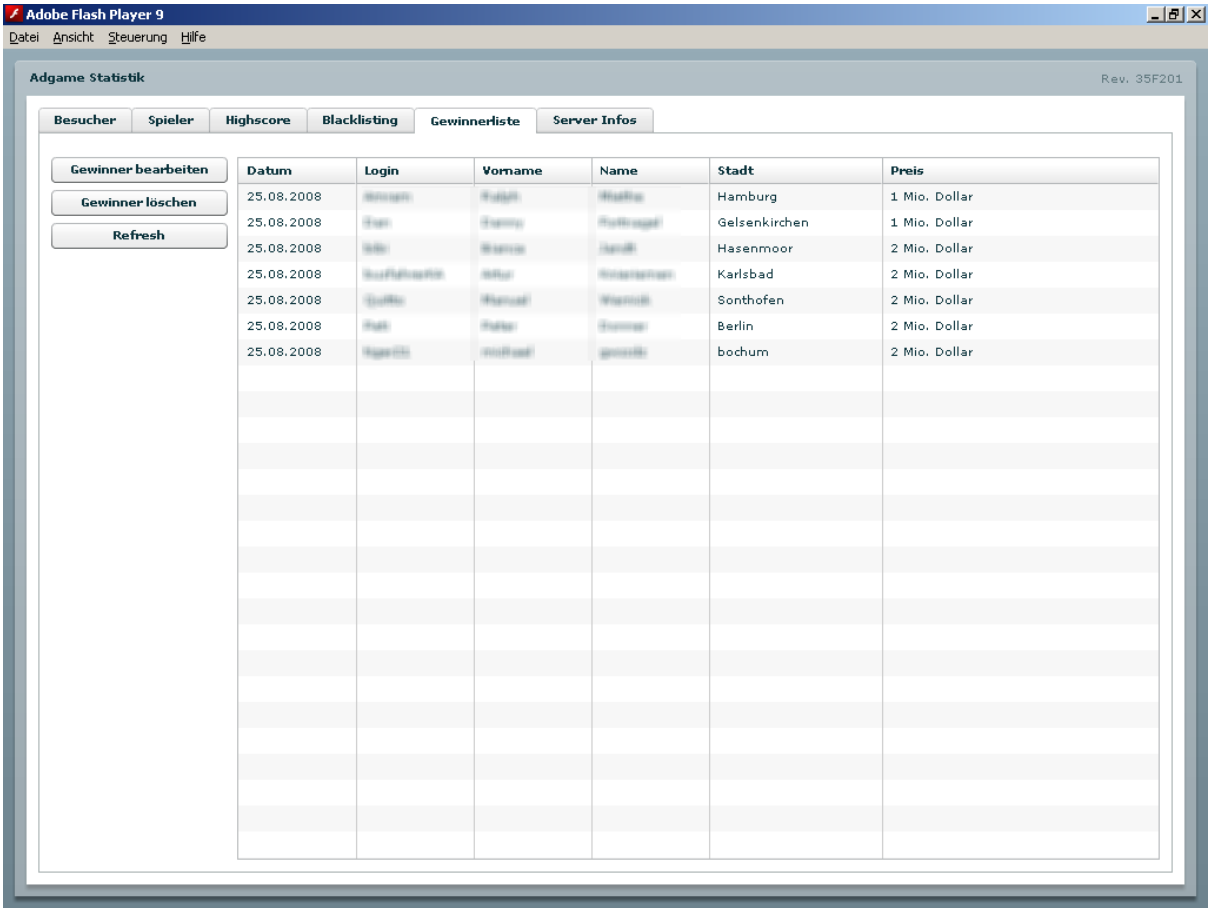
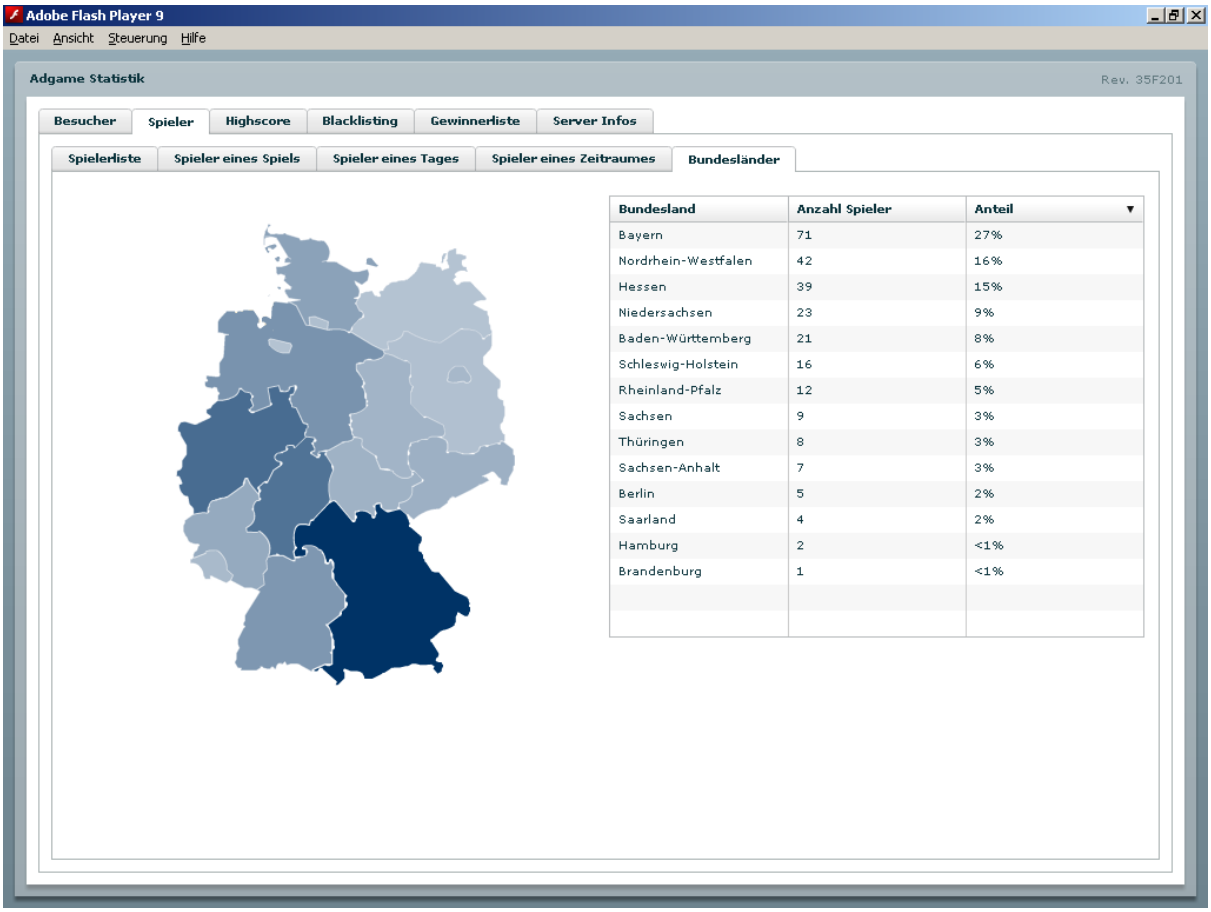


Abbildung 5.12: Liste von registrierten Spielern, die als Gewinner deklariert wurden

5.5.12 Modul PlayerState

In einigen Adgames ist es möglich bzw. sogar nötig, dass ein Spieler die Postleitzahl seines Wohnortes angibt, bevor er ein Spiel spielen kann. Im Modul *PlayerState* wird die Anzahl der registrierten Spieler eines Bundeslandes ausgewertet. Die Daten werden zum Einen graphisch, mittels einer Deutschlandkarte, ausgewertet. Die Bundesländer der Karte werden dazu je nach Anzahl der Spieler je Bundesland im Verhältnis zur Gesamtanzahl der Spieler ganz Deutschlands eingefärbt. Zusätzlich werden die Daten in Tabellenform aufgelistet, so dass auch eine quantitative Aussage über die Spieler der Bundesländer getroffen werden kann.



Bundesland	Anzahl Spieler	Anteil
Bayern	71	27%
Nordrhein-Westfalen	42	16%
Hessen	39	15%
Niedersachsen	23	9%
Baden-Württemberg	21	8%
Schleswig-Holstein	16	6%
Rheinland-Pfalz	12	5%
Sachsen	9	3%
Thüringen	8	3%
Sachsen-Anhalt	7	3%
Berlin	5	2%
Saarland	4	2%
Hamburg	2	<1%
Brandenburg	1	<1%

Abbildung 5.13: Darstellung der Deutschlandkarte und der Länder. Bundesländer je nach Anzahl der Spieler eingefärbt. Die dargestellten Werte in Tabellenform neben der Karte.

5.5.13 Modul ServerStat

Dieses Modul zeigt verschiedenste Informationen über den eingesetzten Server. Typische Infos sind die installierten Versionen der Script-Sprache PHP oder des Adgame-Services. Auch der eingesetzte Web-Server oder das laufende Betriebssystem werden hier angezeigt.

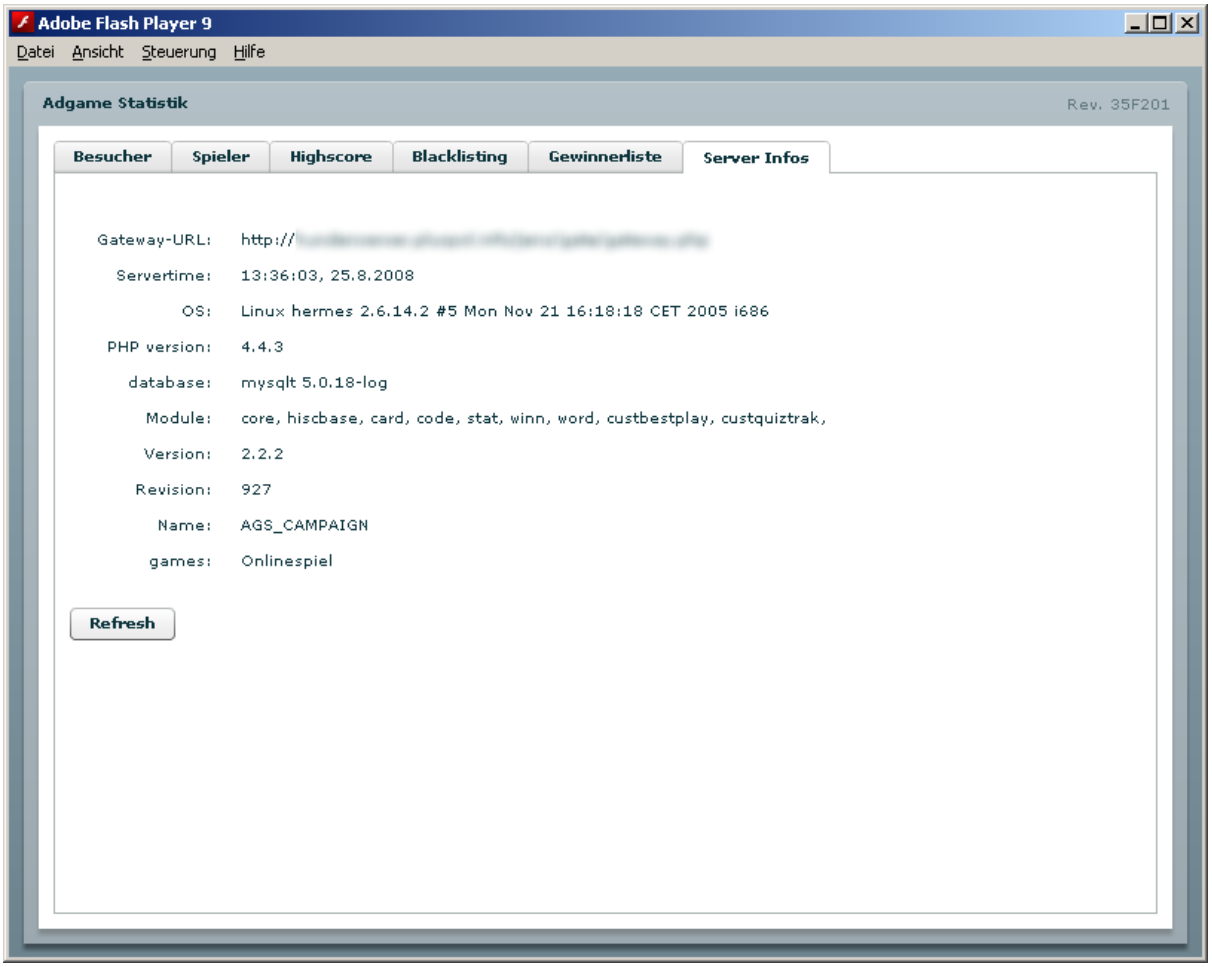


Abbildung 5.14: Serverinformationen

6 Zusammenfassung

Ausgehend von der Aufgabenstellung werden in Kapitel 2 die grundlegende Begriffe erläutert, die für das Verständnis der nachfolgenden Abschnitte wichtig sind. Vor allem die Klärung der Begriffe Application Framework und Rich-Internet-Application sind an dieser Stelle von größter Bedeutung.

Anschließend werden fünf Rich-Internet-Application-Frameworks untersucht. Einige der betrachteten Technologien, wie Java Applets und Flash Applikationen, sind bereits fest im Internet etabliert. Die Java Applet Technologie zeichnet sich besonders durch die hohe Stabilität der JVM-Laufzeitumgebung und eine große Vielfalt an fertigen Paketen und Bibliotheken aus. Da aber Java Applets im Vergleich zu anderen RIA-Technologien recht groß sind, kommen sie weniger im Internet sondern vor allem in Firmennetzwerken zum Einsatz. Die im Internet am weitesten verbreitete RIA-Technologie ist Flash. Allein für die Erstellung von Werbebannern und kleinen Browserspielen gibt es derzeit kaum eine Alternative zu Flash. Die Technologie glänzt besonders im Hinblick auf die Dateigröße der Anwendung, die einfach zu bedienende Flash IDE und eine hervorragende Verbreitung des Flash Players als Laufzeitumgebung. Eine weitere Technologie, die auf das Flash-Format setzt, ist OpenLaszlo. Mit Hilfe dieser OpenSource Software lassen sich – ausgehend vom gleichen Quellcode – nicht nur Flash-Anwendungen compilieren, sondern auch Ajax-Anwendungen exportieren. Ajax ist die vierte in dieser Arbeit vorgestellte RIA-Technologie, die im Gegensatz zu den anderen keine eigene Laufzeitumgebung benötigt und damit die weiteste Verbreitung im Internet vorweisen kann. Microsoft Silverlight ist die letzte und zugleich jüngste Technologie, die im Rahmen dieser Arbeit betrachtet wird. Silverlight benötigt wie Adobe Flash eine eigene Laufzeitumgebung und gilt als dessen direkter Konkurrent. Ob sich diese Technologie jedoch gegen Flash durchsetzen wird, ist zum Zeitpunkt der Erstellung dieser Arbeit noch nicht abzusehen.

Im Hauptteil dieser Arbeit (Kapitel 4) wird explizit das RIA-Framework Flex 3 von Adobe vorgestellt. Zunächst wird allgemein auf das Flex-Framework eingegangen und dessen Besonderheiten, wie die Plattformunabhängigkeit und die gute Dokumentation des SDK dargelegt. Mit dem Bedarf einer Laufzeitumgebung und der geringen Suchmaschinenfreundlichkeit werden an dieser Stelle auch einige Nachteile von Flex aufgezählt. Weiterhin wird die Flex-Programmiersprache MXML und die im Flex 3 SDK integrierten ActionScript-3-Bibliotheken untersucht. Dazu wird ein kleiner Einblick in den

Umfang der visuellen Komponenten des Flex SDK gewährt. Mit der Analyse des SDK wird der immense Umfang des Flex 3 Frameworks offen gelegt. Allein die Flex-3-Bibliothek umfasst über 600 ActionScript-Klassen. Außerdem bieten die zahlreichen Tools und die gute Dokumentation des Frameworks kaum Kritikpunkte für Flex-Programmierer.

Im Rahmen dieser Arbeit wurde ein Programm zur Auswertung von statistischen Daten von Werbespielen mit Flex 3 vom Autor entwickelt. Die Programmierung der Applikation ist im fünften und letzten Kapitel dokumentiert und gewährt einen kleinen Einblick in die praktische Arbeit mit Flex 3. Die Anforderungen an die Applikation werden in Abschnitt 5.2 aufgelistet. Die mit dem RIA-Framework Flex erstellte Anwendung war natürlich schon automatisch eine webbasierte Anwendung. Weiterhin war es mit den zahlreichen Flex-GUI-Komponenten und deren Standardskins ohne großen Programmieraufwand möglich, eine optisch aufgeräumte Programmoberfläche zu erstellen. Da die Werbespiele, von denen die Statistikdaten erfasst werden, genau wie das Statistikprogramm auf Flash basieren, kam es auch bei der Serveranbindung zu keinerlei Problemen. Für die Visualisierung der Daten kam die Flex-Charting-Bibliothek zum Einsatz, die Dank ihrer flexibel konfigurierbaren Komponenten den Ansprüchen vollkommen genügte. Die Verwendung von Flex 3 für die Umsetzung erwies sich somit als sehr vorteilhaft, da diese Technologie schon von vornherein einige der Software-Anforderungen erfüllte .

Glossar

AIR	Adobe Integrated Runtime
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface; englisch für Programmierschnittstelle
AS3	ActionScript 3
AVM1	ActionScript-Virtual-Machine 1
CERN	Conseil Européen pour la Recherche Nucléaire; französisch: Europäische Organisation für Kernforschung
CSS	Cascading Stylesheets
Framework	engl.: Gerüst, Rahmen
GPL	Common Public License
GUI	General User Interface, Programm-Benutzeroberfläche
HDTV	High Definition Television; engl.: hochauflösendes Fernsehen
HQL	Hibernate Query Language; SQL-ähnliche Scriptsprache für Datenbankanfragen
HTML	Hypertext Markup Language
J2SE	Java 2 Standard Edition; Sammlung von Java-APIs
JIT	Just-In-Time-Compiler; wandelt zur Laufzeit ActionScript 3 Bytecode in nativen Flash Player Code um.
JVM	Java Virtual Machine; Laufzeitumgebung für Java-Anwendungen
LPS	Laszlo Presentation Server
MXML	Flex MarkUp Language
NAB	National Association of Broadcasters
Pivot Tabelle	Eine Pivot Tabelle ist eine interaktive Tabelle, die eine Menge gleichartiger Datensätze in Gruppen zusammenfasst. Nach welchen Feldern gruppiert wird und welche Felder ausgegeben werden, kann der Benutzer meist frei wählen.
Refactoring	engl.: Refaktorisierung, Refaktorierung oder schlicht Umgestaltung; bezeichnet in der Software-Entwicklung die manuelle oder automatisierte Strukturverbesserung von Programm-Quelltexten unter Beibehaltung des beobachtbaren Programm-Verhaltens. Dabei

	sollen die Lesbarkeit, Verständlichkeit, Wartbarkeit und Erweiterbarkeit verbessert werden, mit dem Ziel, den jeweiligen Aufwand für Fehleranalyse und funktionale Erweiterungen deutlich zu senken.
Screenreader	Ein Screenreader ist eine Software, die den Inhalt einer Website analysiert und die gefundenen Text dem Benutzer laut vorliest. Sehbehinderte Menschen sind meist auf diese Technologie angewiesen.
SDK	Software Development Kit; Sammlung von Programmen und Dokumentionen, die die Erstellung von eigenen Programmen erleichern/ermöglichen soll.
SOAP	Simple Object Access Protocol; plattformunabhängiges Kommunikationsprotokoll
SQL	Structured Query Language; Datenbanksprache zur Definition, Abfrage und Manipulation von Daten in relationalen Datenbanken
SVG	Scalable Vector Graphics
VM1	(ActionScript-)Virtual-Machine 1
W3C	World Wide Web Consortium; Gremium zur Standardisierung der das World Wide Web betreffenden Techniken
WPF/E	Windows Presentation Foundation/Everywhere
WS-I	Web Services Interoperability Organization
WSDL	Webservice Description Language; plattform-, programmiersprachen- und protokollunabhängige Beschreibungssprache für Netzwerkdienste zum Austausch von Nachrichten auf Basis von XML
WWW	World Wide Web
WYSIWYG	What You See Is What You Get (Was du siehst, ist [das,] was du bekommst.)
XAML	eXtensible Application Markup Language
XML	Extensible Markup Language

Literaturverzeichnis

[Abr07] Abrams, Krzysztof Cwalina; B.: *Richtlinien für das Framework-design*. Addison-Wesley, München, 2007. – 342 S. – ISBN 978-3827324542

[Adoa] Adobe Flex Examples Blog: *Flex Examples Blog*. http://blog.flexexamples.com/, Abruf: 27. Februar 2008. Internet

[Adob] Adobe Flex Support Forums: *Adobe Flex Forum*. http://www.adobe.com/cfusion/webforums/forum/index.cfm?forumid=60, Abruf: 7. Juni 2008. Internet

[Adoc] Adobe Open Source: *Flex 4 - Flex SDK - Confluence*. http://opensource.adobe.com/wiki/display/flexsdk/Flex+4, Abruf: 31. Juni 2008. Internet

[Adod] Adobe Systems Inc.: *Adobe Flex Charting*. http://www.adobe.com/de/products/flex/charting/, Abruf: 9. März 2009. Internet

[Adoe] Adobe Systems Inc.: *Adobe Flex Language Reference*. http://livedocs.adobe.com/flex/3/langref/, Abruf: 9. März 2009. Internet

[Adof] Adobe Systems Inc.: *Adobe to Open Source Flex*. http://www.adobe.com/aboutadobe/pressroom/pressreleases/200704/042607Flex.html, Abruf: 8. März 2009. Pressemitteilung vom 26. April 2007

[Adog] Adobe Systems Inc.: *Adobe verbessert Suche in Rich Media-Inhalten*. http://www.adobe.com/de/aboutadobe/pressroom/pr/jul2008/044.pdf, Abruf: 9. März 2009. Pressemitteilung

[Adoh] Adobe Systems Inc.: *AMF0 Specification*. http://download.macromedia.com/pub/labs/amf/amf0_spec_121207.pdf, Abruf: 8. März 2008. Internet

[Adoi] Adobe Systems Inc.: *AMF3 Specification*. http://download.macromedia.com/pub/labs/amf/amf3_spec_121207.pdf, Abruf: 8. März 2008. Internet

[Adoj] Adobe Systems Inc.: *Flex Compiler Shell*. http://labs.adobe.com/wiki/index.php/Flex_Compiler_Shell, Abruf: 8. März 2008. Internet

[Adok] Adobe Systems Inc.: *Flex FAQ*. http://www.adobe.com/de/products/flashplayer/productinfo/faq/#item-1-5, Abruf: 1. Juni 2008. Internet

[Adol] Adobe Systems Inc.: *SWF searchability FAQ*. http://www.adobe.com/devnet/flashplayer/articles/swf_searchability.html, Abruf: 9. März 2009. Internet

[And08] Anderson, Joshua Noble; T.: *Flex 3 Cookbook*. O'Reilly Media, Inc., 2008. – 704 S. – ISBN 978-0-596-52985-7

[AS05] Andrew Stellman, Jennifer G.: *Applied Software Project Management*. O'Reilly Media, 2005. – 322 S. – ISBN 978-0596009489

[ATH08] Anthony T. Holdener, III: *Ajax: The Definitive Guide*. O'Reilly, 2008. – 980 S. – ISBN 0-596-52838-8

[Bes] Besim Karadeniz: *Geschichte des Internets*. http://www.netplanet.org/geschichte/neunziger.shtml, Abruf: 8. März 2008. Internet

[CF07] Chris Ford, Sanjiv P. Ido Gileadi G. Ido Gileadi: *Patterns for Performance and Operability: Building and Testing Enterprise Software*. Auerbach Pubn, 2007. – 344 S. – ISBN 978-1420053340

[Col08] Cole, Alaric: *Learning Flex 3*. O'Reilly Media, Inc., 2008. – 283 S. – ISBN 978-0-596-51732-8

[Dob08] Dobler, Gustav Pomberger; H.: *Algorithmen und Datenstrukturen: Eine systematische Einführung in die Programmierung*. Pearson Studium, 2008. – 576 S. – ISBN 978-3827372680

[fla] flashmagazin.com: *Flash History*. http://www.flashmagazine.com/news/detail/the_flash_history, Abruf: 8. März 2008. Internet

[Goo] Google Inc.: *Google Maps*. http://maps.google.de/, Abruf: 8. März 2008. Internet

[HMD08] Harvey M. Deitel, Paul J. D.: *AJAX, Rich Internet Applications, and Web Development for Programmers*. Prentice Hall International, 2008. – 1040 S. – ISBN 978-0131587380

[Jes] Jesse James Garrett: *Ajax: A New Approach to Web Applications*. http://www.adaptivepath.com/ideas/essays/archives/000385.php, Abruf: 20. März 2008. Internet

[Jon] Jonathan Gay: *The History of Flash*. http://www.adobe.com/macromedia/events/john_gay/, Abruf: 8. März 2008. Internet

[JT08] Jeff Tapper, Matthew Boles James T. Michael Labriola L. Michael Labriola: *Adobe Flex 3: Training from the Source*. Addison-Wesley Longman, Amsterdam, 2008. – 696 S. – ISBN 978-0321529183

[Kud07] Kudrass, Thomas: *Taschenbuch Datenbanken*. Hanser Fachbuch, 2007. – 582 S. – ISBN 978-3446409446

[Lam07] Lammenett, Erwin: *TYPO3 Online-Marketing-Guide: Affiliate- und E-Mail-Marketing, Keyword-Advertising, Suchmaschinen-Optimierung mit TYPO3*. Gabler, 2007. – 184 S. – ISBN 978-3834906397

[Las] Laszlo Systems Inc.: *OpenLaszlo*. `http://www.openlaszlo.org/`, Abruf: 8. März 2008. Internet

[Lot07] Lott, Chafic Kazoun; J.: *Programming Flex 2*. O'Reilly Media, Inc., 2007. – 504 S. – ISBN 0-596-52689-X

[MC07] Mike Chambers, Jeff S. Rob Dixon D. Rob Dixon: *Apollo for Adobe Flex Developers Pocket Guide*. O'Reilly Media, Inc., 2007. – 144 S. – ISBN 0-596-51391-7

[Mica] Microsoft: *Microsoft Silverlight*. `http://www.microsoft.com/silverlight/`, Abruf: 8. März 2008. Internet

[Micb] Microsoft Corporation: *.NET Framework Developer Center*. `http://msdn.microsoft.com/en-us/library/ms752347.aspx`, Abruf: 9. März 2009. Internet

[Muc03] Muck, Tom: *Flash Remoting*. O'Reilly Media, 2003. – 640 S. – ISBN 978-0596004019

[Nob07] Noble, Roger Braunstein; Mims H. Wright; Joshua J.: *Apollo for Adobe Flex Developers Pocket Guide*. Wiley Publishing, Inc., 2007. – 735 S. – ISBN 978-0-470-13560-0

[Nov] Novell: *Mono*. `http://www.mono-project.com/Main_Page`, Abruf: 8. März 2008. Internet

[RO08] Richard Oates, Stefan Wille Torsten Lueckow Gerald B. Thomas Langer L. Thomas Langer: *Spring & Hibernate. Eine praxisbezogene Einführung*. Hanser Fachbuch, 2008. – 314 S. – ISBN 978-3446412132

[San07] Sanders, William B.: *Learning Flash Media Server 2*. O'Reilly Media, 2007. – 174 S. – ISBN 978-0596510411

[Sca] ScaleNine LLC.: *Flex Builder Feature Creeping*. `http://scalenine.com/blog/2007/03/02/flex-builder-feature-creeping/`, Abruf: 9. März 2009. Pressemitteilung

[Sil07] Silberberger, Holger: *Collaborative Business und Web Services: Ein Managementleitfaden in Zeiten technologischen Wandels*. Springer, Berlin, 2007. – 166 S. – ISBN 978-3540004172

[Ste07] Stein, Erich: *Taschenbuch Rechnernetze und Internet*. Hanser Fachbuch, 2007. – 598 S. – ISBN 978-3446409767

[Sun] Sun Microsystems: *Applets*. http://java.sun.com/applets/, Abruf: 8. März 2008. Internet

[TU-] TU-Wien: *Client-Server-Architektur*. http://gd.tuwien.ac.at/study/hrh-glossar/1-2_17.htm, Abruf: 10. August 2008. Internet

[W3Ca] W3C: *The original proposal of the WWW, HTMLLized*. http://www.w3.org/History/1989/proposal.html, Abruf: 8. März 2008. Internet

[W3Cb] W3C: *Web Services Activity*. http://www.w3.org/2002/ws/, Abruf: 8. März 2008. Internet

[Woy06] Woychowsky, Edmond: *Ajax: Creating Web Pages with Asynchronous JavaScript and XML*. Prentice Hall, 2006. – 432 S. – ISBN 0-13-227267-9

[wwwa] www.at-web.de: *Google liest Flash nun besser, Yahoo! will folgen*. http://www.at-web.de/blog/20080705, Abruf: 9. März 2009. Internet

[wwwb] www.flashdevelop.org: *Flash Develop*. http://www.flashdevelop.org/, Abruf: 9. März 2009. Pressemitteilung

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Alle Teile, die wörtlich oder sinngemäß einer Veröffentlichung entstammen, sind als solche kenntlich gemacht.

Die Arbeit wurde noch nicht veröffentlicht oder einer anderen Prüfungsbehörde vorgelegt.

Mittweida, den 26. April 2009